



**CYBER 170 COMPUTER SYSTEMS  
MODELS 845 AND 855**

**CYBER 180 COMPUTER SYSTEMS  
MODELS 840, 845, 850, 855, AND 860**

**CYBER 840A, 850A, AND 860A  
COMPUTER SYSTEMS**

**CENTRAL PROCESSOR AND CENTRAL MEMORY**

**THEORY OF OPERATION**

---

**HARDWARE MAINTENANCE MANUAL**

# REVISION RECORD

REVISION	DESCRIPTION
A (10-04-82)	Manual released.
B (12-10-82)	Manual revised; includes Engineering Change Order 44142. Pages 2 through 6, 8, 10, 1-2, 1-10, 2-13, 2-14, 3-1/3-2, 3-5, 3-21, 3-25, 3-26, 3-63 through 3-66, 4-1, 4-5, 4-8 through 4-12, 4-15 through 4-17, 4-20, 4-21, 4-39, 4-44 through 4-47, 4-52, 4-53, 6-6, and 6-14 are revised.
C (01-31-84)	Manual revised; includes Engineering Change Order 45258. Front Cover through 8, 16, 1-1, 1-4, 1-6, 1-9, 1-10, 2-2, 2-3, 2-7, 2-29, 2-44, 2-63, 3-7, 3-11, 3-12, 3-14, 3-16, 3-17, 3-19/3-20, 3-24, 3-32, 3-73, 4-7, 4-13, 4-17, 4-18, 4-20, 4-25, 4-36, 4-44, 4-58, 5-19, 5-37, 6-4, and 6-18 are revised.
D (01-30-85)	Manual revised; includes Engineering Change Order 46651. Front Cover through 10, 14 through 17, 1-1, 1-11, 1-12, 2-22, 3-25 through 3-28, 3-42 through 3-47, 5-27 through 5-29, 5-39 through 5-41, 5-43 through 5-69, 8-1, 8-2 are revised.
E (06-13-86)	Manual revised; includes Engineering Change Order 47774. Front Cover through 7, 1-1, 2-22, 3-25, and 5-68 are revised. Page 8.1/8.2 is added.
F (04-03-87)	Manual revised; includes Engineering Change Order 48575. Front Cover, 1-1, 1-12, 5-16, 5-17, 5-37, and 5-38 are revised. Performance Monitoring (TAB), (Divider), and 7-1 through 7-12 are deleted. Section eight has changed to new section seven.
Publication No. 60458170	

REVISION LETTERS I, O, Q, S, X AND Z ARE NOT USED.

Address comments concerning this manual to:

Control Data  
Technical Publications  
4201 N. Lexington Avenue  
St. Paul, MN 55126-9983

or use Comment Sheet in the back of this manual.

© 1982, 1984, 1985, 1986, 1987  
by Control Data  
All rights reserved  
Printed in the United States of America

# MANUAL TO EQUIPMENT LEVEL CORRELATION SHEET

This manual reflects the equipment configurations listed below.

**EXPLANATION:** Locate the equipment type and series number, as shown on the equipment FCO log, in the list below. Immediately to the right of the series number is an FCO number. If that number and all of the numbers underneath it match all of the numbers on the equipment FCO log, then this manual accurately reflects the equipment.

EQUIPMENT TYPE	SERIES	WITH FCOs	COMMENTS
AD112-A	01	-	Released
	02	44015	
	03	44122	
	04	44123	
	05	44465	
	06	44479	
	07	44645	
	08	44798	
	09	44663	
	10	44806	
	11	44180	
	12	44872	
	13	44916	
	14	44921	
	15	44940	
	16	45127	
	17	44805	
	18	45250	
	19	45288	
	20	45142	
	21	45238	
	22	45344	
	23	45797	
	24	ECO45793	
	25	46398	
	26	45341	
	27	46492	
	28	46681	
	29	ECO46082	
	30	46807	
	31	46938	
	32	47130	
	33	ECO46115	
	34	46111	
	35	47415	
	36	47367	
	37	47668	
	38	47865	
	39	ECO48443	
AD112-B	01	-	Released
	02	45257	
	03	45288	
	04	45142	
	05	45238	
	06	45784	
	07	ECO45793	
	08	ECO45341	
	09	46681	
	10	46938	

EQUIPMENT TYPE	SERIES	WITH FCOs	COMMENTS
AD112-C	11	47130	Released
	12	ECO46115	
	13	46111	
	14	47415	
	15	47668	
	01	-	
AD113-A	02	46938	Released
	03	47130	
	04	47415	
	05	47668	
	06	47332	
	01	-	
AT464-A	02	46938	Released
	03	47130	
	04	47576	
	05	47415	
	06	47668	
	01	-	
AT465-A	02	45345	Released
	03	45798	
	04	46397	
	05	46491	
	06	ECO46082	
	07	46806	
AT487-A	08	47366	Released
	09	47860	
	10	ECO48442	
	01	-	
	02	45127	
	03	45344	
AT487-B	04	45797	Released
	05	46398	
	06	46492	
	07	ECO46082	
	08	46807	
	09	47367	
AT487-C	10	47865	Released
	11	ECO48443	
	01	-	
	02	46806	
	03	47366	
	04	ECO47706	
BS137-A	05	47860	Released
	06	ECO48442	
	01	-	
	02	46807	
	03	47367	
	04	ECO47707	
BS137-A	05	47865	Released
	06	ECO48443	
	01	-	
	02	46805	
	03	47365	
	04	ECO47705	
BS137-A	05	47861	Released
	06	48441	
	01	-	
BS137-A	02	ECO41789	Released
	03	42783	
	01	-	



EQUIPMENT TYPE	SERIES	WITH FCOs	COMMENTS
	04	43538	
	05	44583	
	06	45200	
	07	44805	
	08	44680	
	09	48015	
BS145-A	01	-	Released
	02	39845	
BS149-A	01	-	Released
	02	39845	
BS213-A	01	-	Released
	02	47021	
BS214-A	01	-	Released
BS215-A	01	-	Released
BT316-A	01	-	Released



# LIST OF EFFECTIVE PAGES

New features, as well as changes, deletions, and additions to information in this manual, are indicated by bars in the margins or by a dot near the page number if the entire page is affected. A bar by the page number indicates pagination rather than content has changed.

PAGE	REV	PAGE	REV	PAGE	REV	PAGE	REV	PAGE	REV
Front Cover	-	2-20	A	3-7	C	3-65	B	4-31	A
Title Page	-	2-21	A	3-8	A	3-66	B	4-32	A
2	F	2-22	E	3-9	A	3-67	A	4-33	A
3	F	2-23	A	3-10	A	3-68	A	4-34	A
4	F	2-24	A	3-11	C	3-69	A	4-35	A
4.1/4.2	E	2-25	A	3-12	C	3-70	A	4-36	C
5	F	2-26	A	3-13	A	3-71	A	4-37	A
6	F	2-27	A	3-14	C	3-72	A	4-38	A
7	F	2-28	A	3-15	A	3-73	C	4-39	B
8	D	2-29	C	3-16	C	3-74	A	4-40	A
8.1/8.2	F	2-30	A	3-17	C	3-75	A	4-41	A
9	F	2-31	A	3-18	A	3-76	A	4-42	A
10	D	2-32	A	3-19/3-20	C	3-77	A	4-43	A
11	A	2-33	A	Tab	-	3-78	A	4-44	C
12	A	2-34	A	Divider	-	3-79	A	4-45	B
13	A	2-35	A	3-21	B	3-80	A	4-46	B
14	D	2-36	A	3-22	A	3-81	A	4-47	B
15	F	2-37	A	3-23	A	3-82	A	4-48	A
16	F	2-38	A	3-24	C	3-83	A	4-49	A
17	D	2-39	A	3-25	E	3-84	A	4-50	A
Tab	-	2-40	A	3-26	D	3-85	A	4-51	A
Divider	-	2-41	A	3-27	D	3-86	A	4-52	B
1-1	F	2-42	A	3-28	D	3-87	A	4-53	B
1-2	B	2-43	A	3-29	A	3-88	A	4-54	A
1-3	A	2-44	C	3-30	A	3-89	A	4-55	A
1-4	C	2-45	A	3-31	A	Tab	-	4-56	A
1-5	A	2-46	A	3-32	C	Divider	-	4-57	A
1-6	C	2-47	A	3-33	A	Tab	-	4-58	C
1-7	A	2-48	A	3-34	A	Divider	-	4-59	A
1-8	A	2-49	A	3-35	A	4-1	B	4-60	A
1-9	C	2-50	A	3-36	A	4-2	A	Tab	-
1-10	C	2-51	A	3-37	A	4-3	A	Divider	-
1-11	D	2-52	A	3-38	A	4-4	A	4-61	A
1-12	F	2-53	A	3-39	A	4-5	B	4-62	A
1-13	A	2-54	A	3-40	A	4-6	A	4-63	A
1-14	A	2-55	A	3-41	A	4-7	C	4-64	A
1-15	A	2-56	A	3-42	D	4-8	B	4-65	A
1-16	A	2-57	A	3-43	D	4-9	B	4-66	A
Tab	-	2-58	A	3-44	D	4-10	B	4-67	A
Divider	-	2-59	A	3-45	D	4-11	B	4-68	A
2-1	A	2-60	A	3-46	D	4-12	B	4-69	A
2-2	C	2-61	A	3-47	D	4-13	C	4-70	A
2-3	C	2-62	A	3-48	A	4-14	A	4-71	A
2-4	A	2-63	C	3-49	A	4-15	B	4-72	A
2-5	A	2-64	A	3-50	A	4-16	B	4-73	A
2-6	A	2-65	A	3-51	A	4-17	C	4-74	A
2-7	C	2-66	A	3-52	A	4-18	C	4-75	A
2-8	A	2-67	A	3-53	A	4-19	A	4-76	A
2-9	A	2-68	A	3-54	A	4-20	C	4-77	A
2-10	A	2-69	A	3-55	A	4-21	B	4-78	A
2-11	A	Tab	-	3-56	A	4-22	A	4-79	A
2-12	A	Divider	-	3-57	A	4-23	A	4-80	A
2-13	B	3-1/3-2	B	3-58	A	4-24	A	4-81	A
2-14	B	Tab	-	3-59	A	4-25	C	4-82	A
2-15	A	Divider	-	3-60	A	4-26	A	4-83	A
2-16	A	3-3	A	3-61	A	4-27	A	4-84	A
2-17	A	3-4	A	3-62	A	4-28	A	4-85	A
2-18	A	3-5	B	3-63	B	4-29	A	4-86	A
2-19	A	3-6	A	3-64	B	4-30	A	4-87	A

PAGE	REV	PAGE	REV	PAGE	REV	PAGE	REV	PAGE	REV
4-88	A	4-157	A	5-66	D				
4-89	A	4-158	A	5-67	D				
4-90	A	4-159	A	5-68	E				
4-91	A	4-160	A	5-69	D				
4-92	A	Tab	-	Tab	-				
4-93	A	Divider	-	Divider	-				
4-94	A	5-1	A	6-1	A				
4-95	A	5-2	A	6-2	A				
4-96	A	5-3	A	6-3	A				
4-97	A	5-4	A	6-4	C				
4-98	A	5-5	A	6-5	A				
4-99	A	5-6	A	6-6	B				
4-100	A	5-7	A	6-7	A				
4-101	A	5-8	A	6-8	A				
4-102	A	5-9	A	6-9	A				
4-103	A	5-10	A	6-10	A				
4-104	A	5-11	A	6-11	A				
Tab	-	5-12	A	6-12	A				
Divider	-	5-13	A	6-13	A				
4-105	A	5-14	A	6-14	B				
4-106	A	5-15	A	6-15	A				
4-107	A	5-16	F	6-16	A				
4-108	A	5-17	F	6-17	A				
4-109	A	5-18	A	6-18	C				
4-110	A	5-19	C	6-19	A				
4-111	A	5-20	A	Tab	-				
4-112	A	5-21	A	Divider	-				
4-113	A	5-22	A	7-1	F				
4-114	A	5-23	A	7-2	F				
4-115	A	5-24	A	Comment Sheet	F				
4-116	A	5-25	A	Back Cover	-				
4-117	A	5-26	A						
4-118	A	5-27	D						
4-119	A	5-28	D						
4-120	A	5-29	D						
4-121	A	5-30	A						
4-122	A	5-31	A						
4-123	A	5-32	A						
4-124	A	5-33	A						
4-125	A	5-34	A						
4-126	A	5-35	A						
4-127	A	5-36	A						
4-128	A	5-37	F						
4-129	A	5-38	F						
4-130	A	5-39	D						
4-131	A	5-40	D						
4-132	A	5-41	D						
4-133	A	5-42	A						
4-134	A	5-43	D						
4-135	A	5-44	D						
4-136	A	5-45	D						
4-137	A	5-46	D						
4-138	A	5-47	D						
4-139	A	5-48	D						
4-140	A	5-49	D						
4-141	A	5-50	D						
4-142	A	5-51	D						
4-143	A	5-52	D						
4-144	A	5-53	D						
4-145	A	5-54	D						
4-146	A	5-55	D						
4-147	A	5-56	D						
4-148	A	5-57	D						
4-149	A	5-58	D						
4-150	A	5-59	D						
4-151	A	5-60	D						
4-152	A	5-61	D						
4-153	A	5-62	D						
4-154	A	5-63	D						
4-155	A	5-64	D						
4-156	A	5-65	D						

## PREFACE

---

This manual contains information to assist customer engineers in the maintenance of the CONTROL DATA® CYBER 170 Models 845 and 855, CYBER 180 Models 840, 845, 850, 855, and 860, and CYBER 840A, 850A, 860A, and 870A Central Processor (CP) and Central Memory (CM). Information for the following equipments are included:

<u>Equipment</u>	<u>Description</u>
AD112-A/B/C	Central Processor/CMC
AD113-A	Second Central Processor
BS137-A	Central Memory
BS213-A	Central Memory
BS214-A	Central Memory
BS215-A	Central Memory
BS145-A	Memory Increment
BS149-A	Memory Increment
BS316-A	Cache Memory Expansion

### CONVENTIONS

In section 4, part 2, three hyphens indicate a series of zeros needed to complete a 16-digit field.

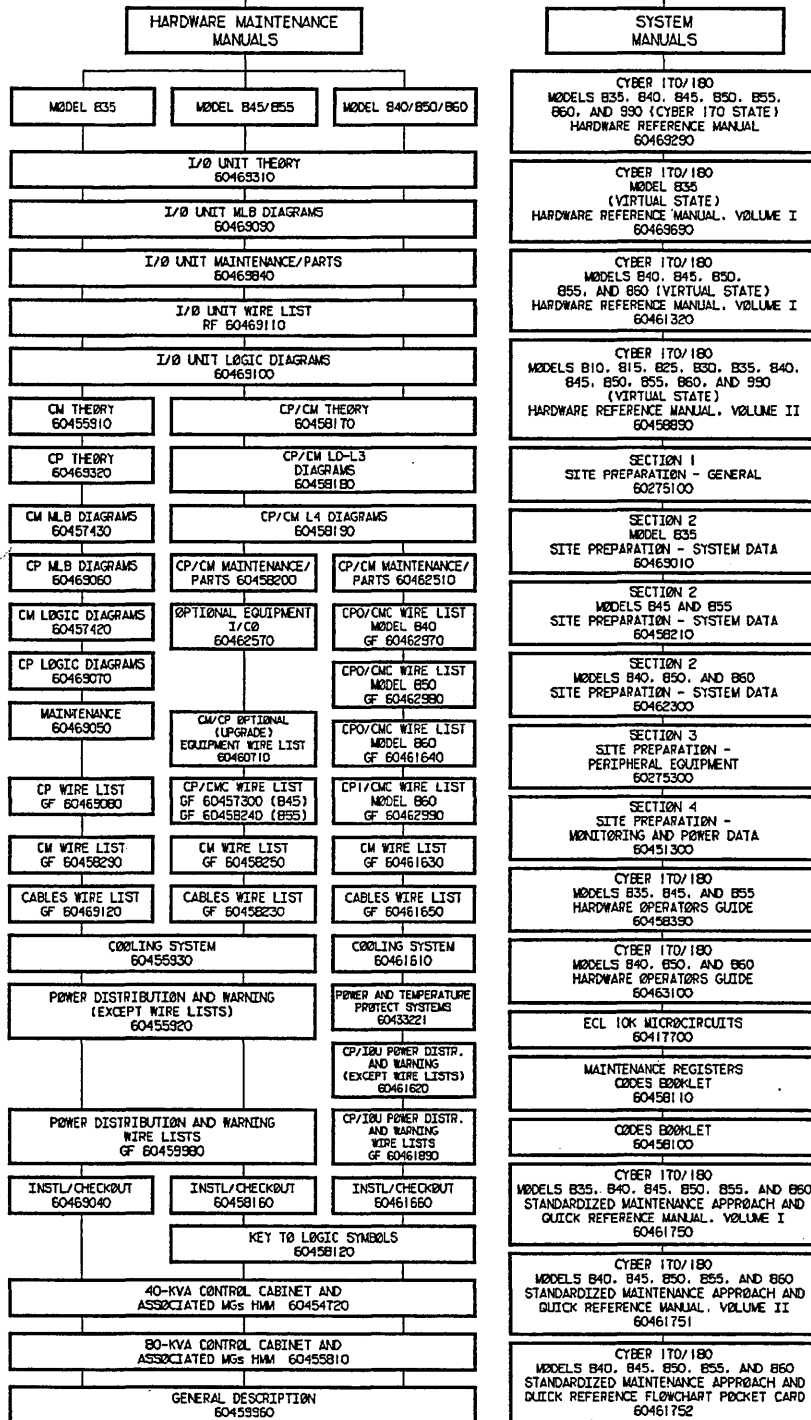
### RELATED PUBLICATIONS

The theory of operation is to be used in conjunction with the CP and CM Level 0 to Level 3 block diagrams contained in the CYBER 170 Models 845 and 855, CYBER 180 Models 840, 845, 850, 855, and 860, and CYBER 840S, 845S, 855S, 840A, 850A, 860A, and 870A Hardware Maintenance Manual, publication number 60458180. Other related publications appear in the System Publication Indexes which follow.

The latest manual revision levels and manual ordering information are available from the Literature Distribution and Services Catalog, publication number 90310500.

# SYSTEM PUBLICATION INDEX

CDC CYBER 170/180  
MODELS 835, 840, 845, 850, 855, AND 860  
HARDWARE MANUALS

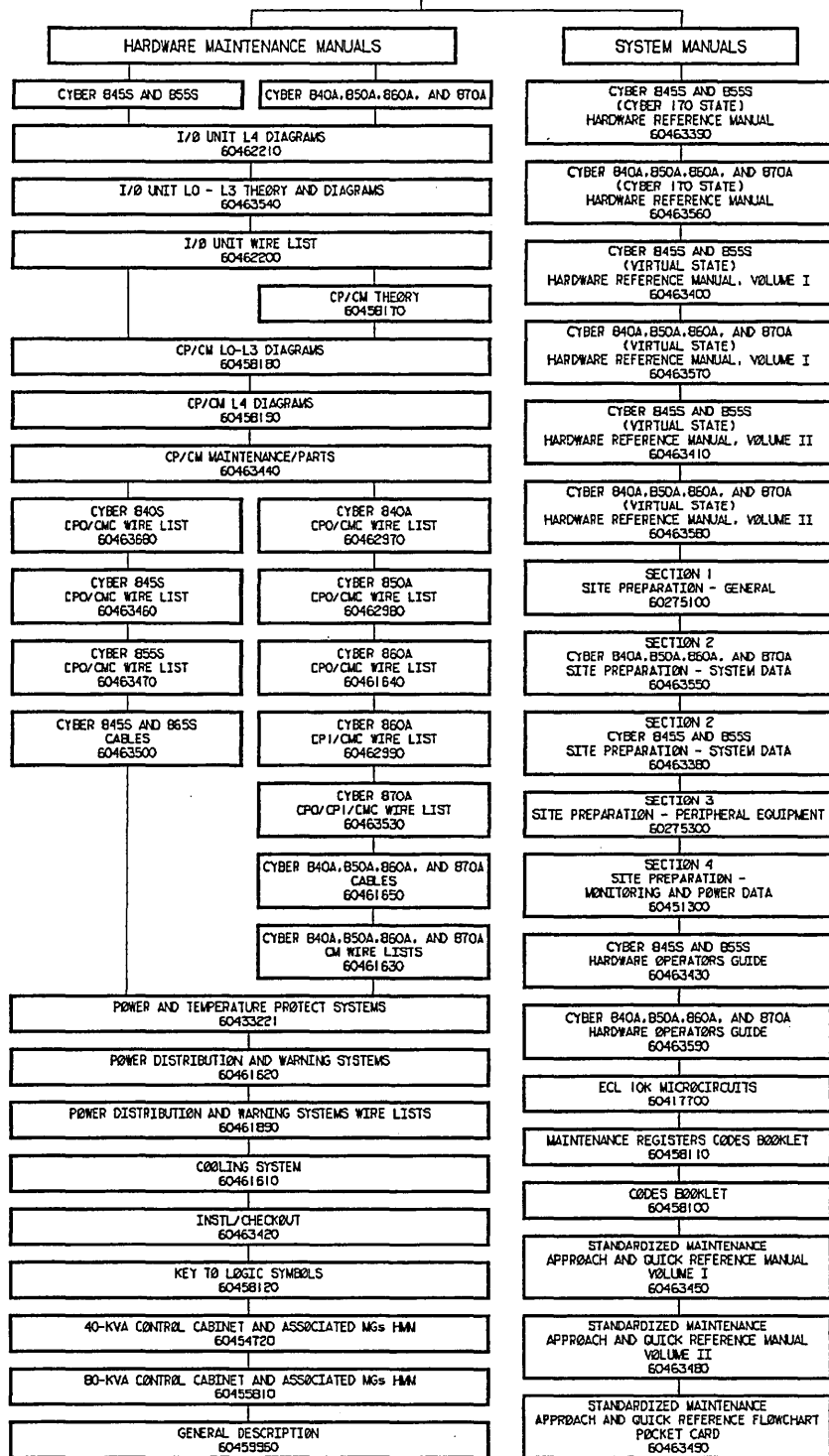


05/65

5132000 851

# SYSTEM PUBLICATION INDEX

CYBER 840S, 845S, 855S, 840A, 850A, 860A, AND 870A  
HARDWARE MANUALS



1/88

SYSTEMS 890 3





# CONTENTS

1. CENTRAL PROCESSOR, CENTRAL MEMORY	1-1	Instruction Execution	1-12
Functional Areas	1-1	Integer Sum, (Xk) Replaced by (Xk) Plus (Xj)	1-12
MAC	1-1	Load Xk From (Aj) Displaced by Q	1-13
IF	1-1	Decimal Sum, D(Ak) Replaced by D(Ak) Plus D(Aj)	1-14
Branch Address Adder	1-2		
Instruction Formatting and Buffering	1-2		
Rank Designators	1-2		
Conditional Branching	1-3		
C170 Mode Address Out of Range (AOR) Detector	1-3		
CST	1-3	2. MAINTENANCE CHANNEL/CP INTERFACE	2-1
Microcode	1-3	Maintenance Access Control (MAC)	2-1
Disassembly Network	1-4	Function Code	2-1
ICP	1-4	Maintenance Channel Protocol	2-4
Rank 13 - Micrand Access	1-4	Function	2-4
Rank 22 - Operand Selection	1-4	Ready Out	2-4
Rank 32 - Functional Area Selection	1-4	Activate Out	2-4
Rank 41 - Functional Unit	1-5	Disconnect Out	2-5
Micrand Execution	1-5	Ready In To IOU	2-5
Rank 50 - Result Selection	1-5	Disconnect In To IOU	2-5
OPI	1-5	Error In To IOU	2-5
Register File	1-5	Status Signals	2-5
DAI Register	1-5	Status Summary	2-5
Live Registers	1-6	Exchange Accept	2-5
ICC	1-6	Processor Fault Status (PFS)	2-5
Microtrap Code	1-6	Dependent Environment Control (DEC)	2-5
Clear Pipe	1-6	Processor Test Mode (PTM)	2-6
ALN	1-6	MAC Operations	2-6
AC	1-7	Function	2-8
Address Formation	1-7	IOU Immediate Operations	2-8
BDP Instructions	1-7	Stop (Function Code = 0X)	2-9
Load/Store Instructions	1-8	Start (Function Code = 1X)	2-9
AC Control	1-8	Master Clear (Function Code = 60 or 6A)	2-10
BDP	1-9	Clear Error (Function Code = 70 or 7A)	2-10
Data Flow	1-9	IOU Read, Write Operations	2-12
Stream Pausing	1-9	Activate Out	2-12
SM and LM	1-9	Ready Out	2-13
Segment Map RAM	1-9	Control Word 1	2-13
Access Validation Testing	1-10	Control Word 2	2-13
Cache Memory	1-10	Read, Write Initial Steps	2-14
Page Map RAM	1-10	Echo (Function Code = 8X)	2-15
CMC Tag	1-10	Direct Reads, Writes	2-15
CMC and CM	1-11	CMC Maintenance Register	
Refresh	1-11	Read, Write	2-16
Write	1-11	CMC Maintenance Register	
Read	1-11	Read (Function Code = 4A)	2-16
Partial Write	1-11	CMC Maintenance Register	
Maintenance Registers	1-11	Write (Function Code = 5A)	2-17

CP Status Summary Read (Function Code = 40, Address = 00)	2-18	MAC-Resident Register Read	2-63
CST Micrand Address Register, Breakpoint Register Read (Function Code = 40, Address = 31, 32)	2-19	MAC-Resident Register Write (PFS, PTM Only)	2-65
CST Micrand Address Register, Breakpoint Register Write, (Function Code = 50, Address = 31, 32)	2-20	PMF Register 21 Write	2-67
CST RAM Read, Write	2-22		
CST RAM Read (Function Code = 41)	2-22	3. INSTRUCTION ISSUE	3-1
CST RAM Write (Function Code = 51)	2-25	Part 1 - Instruction Fetch (IF)	3-3
Control Memory Read, Write	2-27	Branch Instructions	3-3
Reference ROM Write (Function Code = 53)	2-27	Cl70 Mode	3-4
Soft Control Memory Write (Function Code = 54)	2-29	Conditional, Relative	3-4
BDP Memory Write (Function Code = 55)	2-32	Unconditional, Indexed	3-4
First Level Instruction Decoder A, B Write (Function Code = 56)	2-36	Cl80 Mode	3-4
Control Memory Read (Function Codes = 43 through 46)	2-39	Conditional, Relative	3-4
Register File, Read, Write	2-40	Unconditional, Indexed	3-4
Register File Read (Function Code = 47)	2-40	IF Functions	3-5
Register File Write (Function Code = 57)	2-42	Instruction Assembly	3-5
MAC Reads, Writes	2-44	First Level Instruction Decoder	3-10
Element ID, Processor ID, Options Installed Read (Function Code = 40, Address = 10, 11, 12)	2-44	Instruction Formatting, Buffer Word Assembly	3-11
DEC, PTM, PFS Register Read (Function Code = 40, Address = 30, A0, 8X)	2-46	Instruction Buffer Ranks 2, 3, 10, 11, and 12	3-12
DEC, PTM, PFS Register Write (Function Code = 50, Address = 30, A0, 8X)	2-49	P Register	3-12
PMF Register 21 and 22 Read (Function Code = 40, Address = 21, 22)	2-52	Branch Address Adder	3-12
PMF Register 22 Write (Function Code = 50, Address = 22)	2-55	RNI Instruction Request	3-12
Microcode-Assisted Reads, Writes	2-57	Conditional Branch Instruc- tion Request	3-14
Microcode-Assisted Read (Function Code = 40, Ad- dress = 13, 4X-6X, CX or EX)	2-58	Unbranch Instruction Request	3-15
Microcode-Assisted Write (Function Code = 50, Ad- dress = 13, 4X-6X, CX or EX)	2-60	Unconditional Branch Instruc- tion Request	3-16
CP Read, Write Operations	2-63	Branch or Unbranch to Parcel 3 Instruction Word Request	3-18
		Part 2 - Instruction Execution	3-21
		Control Store (CST 1.0)	3-25
		CST RAM	3-25
		Micrand Addressing and Sequence	
		Control (MSC)	3-25
		Entry Addresses	3-26
		Rank 12 Opcode	3-26
		MAC Microcode Address	3-26
		Reference ROM Data	3-26
		Microtrap Code	3-27
		Other CST RAM Entry Addresses	3-27
		CST Opcode	3-28
		General Micrand Register	3-28
		Functional Unit Micrand Registers	3-28
		Disassembly Network	3-28
		Micrand Formats	3-28
		General Micrand	3-31
		AC/LM/SM Micrand	3-42
		BDP Micrand	3-47
		ALN Micrand	3-54

Instruction Control Pipeline (ICP 1.0)	3-62	General Network	4-2
P Registers	3-63	Shift Count Generator	4-3
Immediate Operand, j and k Registers	3-63	Shift Network	4-4
Instruction Control Signal Registers	3-63	Large Adder	4-4
Debug Profile	3-64	Normalize Network	4-4
Largest Ring	3-64	Exponent Arithmetic Network	4-4
General Micrand Registers	3-64	ALN Output Multiplexer	4-5
Pipe Valid Control	3-64	Endcase Control	4-5
Functional Unit Go Response Control	3-65	Branch Condition Control	4-5
Operand Issue (OPI 1.0)	3-65	Multiply/Divide Network	4-5
General Micrand Control	3-66	Multiplication	4-6
OPI Reads, Writes	3-69	B,C Registers	4-8
Read Operations	3-69	Product Networks A, B	4-8
Register Addressing Function (RAF)	3-69	Multiply Partial Summing Network	4-11
Literal Register Address (LRA)	3-69	BDP Decimal-to-Binary Conversion	4-13
Literal Data (LD)	3-70	Multiply Final Adder	4-15
Data Subfunction Select (DSS)	3-70	Multiply/Divide Output Multiplexer	4-15
Live Register Read (LRR)	3-70	Division	4-15
Operand Registers	3-70	C170 Population Counter	4-19
Write Operations	3-71	C170/C180 Differences	4-19
Register Addressing Function (RAF)	3-71	Floating-Point Operations	4-19
Literal Register Address (LRA)	3-72	Floating-Point Exponent Ranges	4-20
Live Register Write/Complete Cycle Control (LRW/CSC)	3-72	Endcase Conditions	4-21
Operand Conflicts, Shortstopping	3-74	Integer/Floating-Point Conversions	4-23
Exchange Operations	3-77	Normalization and Rounding	4-23
Register File Maps	3-83	Operands	4-23
Address Offset	3-88	Conditional Branch Instructions	4-23
Timers	3-89	Instruction Sequences	4-24
Process Interval Timer (PIT)	3-89	C170 Right Shift (21) Instruction	4-24
System Interval Timer (SIT)	3-89	C180 Integer Add (24) Instruction	4-26
Offline MAC Operations	3-89	C180 Single-Precision Floating-Point Sum (30) Instruction	4-28
		C170 Single-Precision Floating-Point Product (40) Instruction	4-38
		General Network Operations	4-46
		C180 Integer Quotient (27) Instruction	4-50
		Quotient Storage	4-56
		Implicit Copy Operation	4-58
		Part 2 - Address Control(AC)	4-61
4. EXECUTING FUNCTIONAL UNITS	4-1	Micrand Control	4-61
Part 1 - Arithmetic and Logical Network (ALN)	4-1	Soft Control	4-61
ALN Control	4-1	Soft Control 1	4-61
ALN Control Field Descriptions	4-2	Soft Control 2	4-63
		Pseudo Soft Control 3	4-64
		Soft Control Address Sequences	4-64
		Store Word	4-64
		Load Bytes	4-64

60458170 A

Data Paths	4-110	Fourth Micrand (#30) -	
A and B Stream Stages 1 through 4	4-110	Move After Convert Aj	
Decimal (Types 0 through 8)	4-110	to Binary	4-132
Alphanumeric (Type 9)	4-110	Decimal Scale (E4jkiD)	4-133
Binary (Type 10, 11)	4-110	Decimal Scale Rounded (E5jkiD)	4-135
Translate (Types 12 through 15)	4-110	Byte Instruction Sequences	4-136
Stage 4	4-111	Move Bytes (76jk)	4-136
ALU and Common Stages 5 through 7	4-111	Byte Compare (77jk)	4-137
Buffer RAM	4-111	Byte Compare Collated (E9jkiD)	4-138
C Stream Stages 1 through 5	4-111	First Micrand (#22)	4-139
Binary/Decimal Converter	4-112	Second Micrand (#3)	4-139
Register Files A and B	4-112	Byte Translate (EBjkiD)	4-140
Register File Compare Control	4-113	First Micrand (#22)	4-140
Scan Hit Control	4-113	Second Micrand (#5)	4-141
Edit	4-113	Edit (EDjkiD)	4-142
Instruction Sequences	4-113	First Micrand (#29)	4-143
BDP Initialization Sequence	4-115	Second Micrand (#23)	4-144
First Micrand (#22)	4-115	Third Micrand (#6)	4-144
Second Micrand (#41)	4-116	Byte Scan While Nonmember	
Third Micrand (#42)	4-116	(F3jkiD)	4-145
Fourth Micrand (#39)	4-116	First Micrand (#35)	4-146
Numeric Instruction Sequences	4-117	Second Micrand (#4)	4-146
Decimal Sum (70jk)	4-117	Calculate Subscript (F4jkiD)	
Decimal Difference (71jk)	4-118	Instruction Sequence	4-147
Decimal Product (72jk)	4-119	First Micrand (#38)	4-148
First Micrand (#24)	4-120	Second Micrand (#37)	4-148
Second Micrand (#40)	4-120	Immediate Data Instruction	
Third Micrand (#26)	4-120	Sequences	4-149
Fourth Micrand (#40)	4-120	Move Immediate Data (F9jkiD)	4-149
Microcoded Binary Multiply	4-121	Compare Immediate Data (FAjkiD)	4-151
Fifth Micrand (#31)	4-121	Add Immediate Data (FBjkiD)	4-152
Sixth Micrand (#28)	4-122	Convert-to-Binary Sequence	4-154
Decimal Quotient (73jk)	4-123	Convert-to-Decimal Sequence	4-155
First Micrand (#25)	4-124	Convert-to-Decimal Network	4-155
Second Micrand (#40)	4-124	First Byte From Buffer	4-158
Third Micrand (#26)	4-124	Second Byte From Buffer	4-159
Fourth Micrand (#40)	4-125	Third Byte From Buffer	4-159
Microcoded Binary Divide	4-125	Converted Decimal to A Stream	4-160
Fifth Micrand (#31)	4-125		
Sixth Micrand (#28)	4-126		
Decimal Compare (74jk)	4-126		
Numeric Move (75jk)	4-128		
First Micrand (#34) -			
All Moves	4-129		
Second Micrand (#11) -			
Standard Move	4-129		
Second Micrand (#27) -			
Move After Convert Aj			
to Decimal	4-130		
Third Micrand (#32) -			
Move After Convert Aj			
to Decimal	4-131		
Second Micrand (#36) -			
Move After Convert Aj			
to Binary	4-131		
Third Micrand (#40) -			
Move After Convert Aj			
to Binary	4-132		
		5. VIRTUAL MEMORY	5-1
		Virtual Memory Address Conversion	5-1
		Basic Conversion Components	5-1
		Segment Table Address and	
		Length Registers	5-1
		Process Segment Table	5-3
		Page Table Address, Length,	
		and Page Size Mask Registers	5-3
		System Page Table	5-3
		PVA to SVA Conversion	5-3
		SVA to RMA Conversion	5-4
		Map Concepts	5-5
		Virtual Memory Address Security	5-5
		Segment Access Privileges	5-6
		Segment Types	5-6

Segment Descriptor Access		Cache Status RAM	5-27
Control Fields	5-6	Cache Parity Checker/Failing	
Execute Privilege (XP)		Chip Indicator	5-27
Field	5-6	Update Control	5-27
Read Privilege (RP) Field	5-7	Allocation Control	5-27
Write Privilege (WP) Field	5-7	Modes of Operation	5-28
Ring Address Protection	5-7	Cache Read	5-28
Ring Hierarchy	5-7	Cache Write	5-28
Accessor Ring Number	5-7	Cache Bypass Read or Write	5-28
Segment Descriptor Ring Fields	5-8	Real Memory Address	5-28
Code Base Pointer Ring Limit	5-8	Six-Byte Write	5-28
Ring Definitions	5-8	Interrupt	5-28
Execute Access	5-8	Stream	5-28
Read/Write Accesses	5-9	Unconditional Look Ahead	5-28
Call Access	5-9	Conditional Look Ahead	5-29
Multiring Segment Example	5-9	Test	5-29
Write	5-10	Prevalidate	5-29
Read	5-10	Index	5-29
Execute	5-10	Purge	5-29
Call	5-11	Invalidate	5-29
Ring Usage Example	5-11	Fake CM	5-29
Key/Lock Address Protection	5-12	Conflicts	5-30
P Register Key Field	5-12	Block Fill/Allocate	5-30
Segment Descriptor Key/Lock		Block Fill/Cache Write	5-30
Fields	5-12	Invalidate	5-30
Key/Lock Applications	5-13	CM Busy	5-30
Key/Lock Usage Example	5-13	Write Followed by Read	5-30
Segment Map	5-15	Allocate/Word Not Valid	5-30
Micrand Control	5-15	Read Access	5-30
PVA to SVA Conversion	5-15	LM Sequences	5-30
Segment Map RAMs	5-15	LM Read Sequence	5-31
PVA Register	5-16	Cache Bypass LM Read Sequence	5-32
Segment Descriptor Input		LM Write Sequence	5-33
Register	5-17	Page Table Search Sequence	5-34
Segment Map RAM Allocation	5-17	Cache Hit and LRU Update	
Segment Descriptor Selection	5-17	Sequences	5-36
Segment Descriptor Fetch Sequence	5-18	Central Memory Control	5-39
P Register (Upper)	5-20	Input Port Interface Signals	5-39
Address Security Tests	5-20	Write Data	5-39
Segment Table Length Test	5-20	Address	5-39
Segment Validity Test	5-20	CMC Mark	5-39
Key/Lock Test	5-20	CMC Tag	5-39
Ring Test	5-21	CMC Function Code	5-40
Local Memory	5-21	Request	5-40
LM Control	5-21	Output Port Interface Signals	5-40
Micrand Control	5-21	CM Read Data	5-40
CPU Requests	5-21	CMC Tag	5-40
Tags and Response	5-22	CMC Response	5-40
Cache Tag File	5-22	CMC Response Code	5-41
SVA Paths	5-22	Interrupt	5-41
SVA to RMA Conversion	5-22	Port Busy	5-41
Page Map RAMs	5-23	Functions	5-41
Page Table Search Control	5-25	Read	5-41
Cache Memory	5-25	Write	5-42
Cache Data RAM	5-25	Read and Set Lock; Read and	
Cache Tag RAM	5-25	Clear Lock; Exchange	5-42
Cache Word Address Valid RAM	5-27	Read and Set Lock	5-42
Cache Tag File Address RAM	5-27	Read and Clear Lock	5-42

Exchange	5-42	Memory Array Size	5-58
Read Free Running Counter	5-42	Bank Size	5-58
Refresh Counter Resync	5-43	Quadrant Size	5-58
Interrupt	5-43	CM Size	5-59
Major Components	5-43	CM Addressing	5-59
Input Ports	5-43	Interface Signals From CMC	5-59
Aux and IOU Buffer Memories	5-43	Go Bank	5-59
Write Error Correction Code		Address	5-59
(ECC) Generator	5-43	Write Data/Write Code, Write	
Conflict and Bank Busy Control	5-43	Parity	5-59
Refresh Address Generator	5-44	Partial Write Data/Partial	
CMC Control and Delay	5-44	Write Code	5-59
Single Error Correction/Double		Write	5-59
Error Detection (SECDED)	5-44	Multiple Bit Error (MBE)	5-60
Partial Write Network	5-44	Two Pass Function	5-60
Partial Write ECC Generator	5-44	Enable Long Cycle	5-60
Output Ports	5-44	Refresh	5-60
Maintenance Registers	5-45	Interface Signals to CMC	5-60
Free Running Counter	5-45	Address PE Bytes 4-7	5-60
Bounds Fault Detector	5-45	Write Data/Write Parity	5-60
Timing	5-45	Read Data/Read Code	5-60
Central Memory (BS137-A)	5-50	Banks 0-7 Clear Busy	5-61
Interface Signals from CMC	5-50	Address Bits 37-42 Inverted	5-61
Address	5-50	Operations	5-61
Go Bank	5-50	Read	5-61
Refresh Request	5-50	Write	5-63
Refresh Request or Inter-		Partial Write	5-65
leaved Mode	5-50	Long Read Cycle	5-68
CM Write	5-50		
Two Pass Function	5-51	6. INSTRUCTION COMPLETION CONTROL	
Multiple Bit Error	5-51	(ICC)	6-1
Enable Long Cycle	5-51	Errors, Exceptions	6-1
Partial Write Data/Code	5-51	CCR, PONR	6-4
Write Data/Parity, Code	5-51	Condition Descriptions	6-6
Interface Signals to CMC	5-51	Interrupt Condition Groups	6-13
Clear Bank Busy	5-51	Error Handling	6-14
Read Data/Code	5-51	Instruction Retry	6-15
Write Data/Parity	5-52	Interrupts	6-15
Memory Organization	5-52	Multiple Interrupt Conditions	6-19
Memory Addressing	5-52		
Functions	5-52	7. CLOCKS	7-1
Read	5-52	Master Clock	7-1
Write	5-52	Clock Distribution	7-1
Partial Write	5-53	Phase Maker	7-1
Long Read Cycle	5-53	Second Stage Clock Fanout	7-1
Partial Write Operation	5-54	Maintenance Features	7-2
Central Memory (BS213-A, BS214-A,			
BS215-A)	5-58		
Physical CM Organization	5-58		
Word Size	5-58		
Chip Size	5-58		

## FIGURES

2-1	MAC Data, Address, Byte Select	2-2	Type Code 3 - Reference ROM
	Buses		Address, Data Formats
2-7		2-29	

2-3	Type Code 4 - Soft Control Memories Address, Data Formats	2-32	4-11	C180 Single-Precision Floating- Point Addition/Subtraction	4-36
2-4	Type Code 5 - BDP Memories Address, Data Formats	2-35	4-12	C180 Double-Precision Floating- Point Sum (34) Instruction	4-37
2-5	Type Code 6 - First Level Instruction Decoder Address, Data Formats	2-38	4-13	Multiply Minor Cycle Control	4-40
3-1	C170 Instruction Words	3-6	4-14	C Register Contents	4-45
3-2	C180 Instruction Words	3-6	4-15	Integer Divide Soft Control	4-52
3-3	C170 Instruction Formats	3-8	4-16	BDP Load Right-to-Left Operation	4-79
3-4	C180 Instruction Formats	3-8	4-17	BDP Store Right-to-Left Operation	4-82
3-5	C180 BDP Instruction Formats	3-9	4-18	BDP Load Binary Data Operation	4-86
3-6	First Level Instruction Decoder Formats	3-10	4-19	Store Word Instruction	4-89
3-7	Instruction Formatting Bit Assignments	3-11	4-20	Load/Store Bit Address Formation	4-91
3-8	Instruction Fetch RNI	3-13	4-21	Load Bit Instruction	4-92
3-9	Conditional Branch (Not to Parcel 3)	3-14	4-22	Store Bit Instruction	4-94
3-10	C180 (90) Branch to P Displaced by 2 Times Q if (Xj) Right = (Xk) Right	3-15	4-23	Test and Set Bit Instruction	4-97
3-11	Unbranch (Not to Parcel 3)	3-16	4-24	Store Bytes Instruction	4-101
3-12	Unconditional Branch (Not to Parcel 3)	3-17	4-25	BDP Initialization Micrands	4-115
3-13	C180 (2F) Branch to P Dis- placed by 2 Times (Xk) Right	3-18	4-26	Decimal Sum Micrands	4-117
3-14	Conditional Branch (to Parcel 3)	3-19	4-27	Decimal Difference Micrands	4-118
3-15	Instruction Unit Operation	3-24	4-28	Decimal Product Micrands	4-119
3-16	General Micrand	3-29	4-29	Decimal Quotient Micrands	4-123
3-17	Functional Unit Micrands	3-30	4-30	Decimal Compare Micrands	4-126
3-18	Integer Sum/Difference (C170 36, 37, C180 24, 25) Instructions	3-67	4-31	Numeric Move Micrands	4-128
3-19	C180 Load Word (82) Instruction	3-68	4-32	Decimal Scale Micrands	4-133
3-20	Operand Access By C180 Integer Sum (24) and C180 Load Word (A2) Instructions	3-73	4-33	Decimal Scale Rounded Micrands	4-135
3-21	Operand Conflict Timing For Three Consecutive Integer Sum Instructions (X1+X2 -> X2, X2+X3 -> X3, X2+X4 -> X4)	3-76	4-34	Move Bytes Micrands	4-136
3-22	C170, C180 Exchange Packages	3-79	4-35	Byte Compare Micrands	4-137
3-23	C170 Register File Map	3-83	4-36	Byte Compare, Collated Micrands	4-138
3-24	C180 Register File Map	3-85	4-37	Byte Translate Micrands	4-140
4-1	Multiplication Network	4-7	4-38	Edit Micrands	4-142
4-2	Demonstration of Booth's Algorithm	4-9	4-39	Byte Scan While Nonmember Micrands	4-145
4-3	Multiply Partial Summing Network Inputs, Outputs	4-12	4-40	Calculate Subscript Micrands	4-147
4-4	Decimal-to-Binary Conversion	4-14	4-41	Move Immediate Data Micrands	4-149
4-5	Integer Division	4-17	4-42	Compare Immediate Data Micrands	4-151
4-6	Floating-Point Division	4-18	4-43	Add Immediate Data Micrands	4-152
4-7	C170 Exponents	4-20	4-44	BDP Convert-to-Decimal Network	4-157
4-8	C180 Exponents	4-20	4-45	Convert-to-Decimal Example Arithmetic	4-158
4-9	C170 Floating-Point Endcase Results	4-21	5-1	Virtual Memory Address Conversion	5-2
4-10	C180 Floating-Point Endcase Results	4-22	5-2	Multiring Segment Example	5-10
			5-3	Ring Usage Example	5-11
			5-4	Key/Lock Usage Example	5-14
			5-5	Segment Map RAM Format	5-16
			5-6	Segment Descriptor Fetch	5-19
			5-7	Page Map RAM Format	5-24
			5-8	Cache Memory	5-26
			5-9	LM Read, Cache Hit, LRU Update	5-37
			5-10	LM Write, Cache Hit, LRU Update	5-38
			5-11	CMC Request/CM Read Data Out Timing (AD112-C)	5-46
			5-12	CMC Request/CM Read Data Out Timing (AD112-A/B)	5-47
			5-13	CMC Timing (AD112-C)	5-48
			5-14	CMC Timing (AD112-A/B)	5-49
			5-15	CM Partial Write	5-55



5-16	CM Bank Timing	5-56	6-1	MCR Conditions	6-2
5-17	CM Read Timing	5-62	6-2	UCR Conditions	6-3
5-18	CM Write Timing	5-64	6-3	CCR, PONR Implementation	6-5
5-19	CM Partial Write Timing	5-67	6-4	Basic Interrupt Mechanism	6-16
5-20	CM Long Read Cycle Timing	5-69			

#### TABLES

2-1	Registers Addressable by Copy State Instruction, MCU	2-2	3-5	ALN Micrand Bits and Descriptions	3-54
3-1	First Level Instruction Decode Bit Definitions	3-10	3-6	C180 Exchange Counter Operations	3-82
3-2	General Micrand Bits and Descriptions	3-31	6-1	MCR Conditions and Descriptions	6-6
3-3	AC/LM/SM Micrand Bits and Descriptions	3-42	6-2	UCR Conditions and Descriptions	6-11
3-4	BDP Micrand Bits and Descriptions	3-47	6-3	Interrupt Condition Groups	6-13
			6-4	PVA Instruction Pointer Function	6-14
			6-5	MCR Interrupts	6-17
			6-6	UCR Interrupts	6-18



## SECTION 1

CENTRAL PROCESSOR, CENTRAL MEMORY



---

The following description of the CYBER 170 Models 845 and 855, CYBER 180 Models 840, 845, 850, 855, and 860, and CYBER 840A, 850A, 860A, and 870A Central Processor (CP) is intended for use with the Level 0 Diagram. Central Memory (CM) and the following major functional areas of the CP appear in the diagram.

- Maintenance Access Control (MAC)
- Instruction Fetch (IF)
- Control Store (CST)
- Instruction Control Pipeline (ICP)
- Operand Issue (OPI)
- Instruction Completion Control (ICC)
- Arithmetic and Logical Network (ALN)
- Address Control (AC)
- Business Data Processor (BDP)
- Segment Map (SM)
- Local Memory (LM)
- Central Memory Control (CMC)

This section includes analysis of all functional areas and major signal paths. Additional text traces the execution of three instructions through the CP.

#### FUNCTIONAL AREAS

##### MAC

MAC is the CP interface with the Maintenance Control Unit (MCU) in the IOU. The Maintenance Channel connects MAC to the MCU.

MAC initializes the CP by loading information required for system operation into Random Access Memories (RAMs). These RAMs are distributed throughout the CP to perform code conversions and provide control facilities. The MAC Data and Address buses connect them to MAC. MAC also provides Dependent Environment Control (DEC) signals to activate performance and maintenance features of the machine, and Processor Test Mode (PTM) signals to force parity errors for maintenance purposes. MAC monitors error signals from all functional areas in the processor, except CM and CMC. It stores the error signals in Processor Fault Status (PFS) registers. MAC indicates the presence of such errors by means of a status summary and supplies PFS data to the MCU for error logging purposes.

Additionally MAC interfaces with PMF, which monitors selected events occurring in the CP. Statistical information gathered by PMF is used to evaluate and improve system performance.

##### IF

IF requests instructions from LM and reformats them for eventual processing in ICP and CST. Instruction buffer ranks in IF minimize delays encountered in other sections of the CP that result from waiting for instructions.

### Branch Address Adder

The Branch Address Adder issues IF Address, the byte number address of an instruction word. IF Address accompanies IF Request, the instruction request, to LM.

Data Interchange (DAI) enters the Branch Address Adder to become the first IF Address of the instruction sequence. P (lower) provides addresses for subsequent instructions in a Read Next Instruction (RNI) sequence. IF Address also may be a branch address.

### Instruction Formatting and Buffering

IF parcels out the instructions contained in the requested LM Read Data word in Instruction Assembly. The instruction words contain from two to four instructions, which in C170 mode are either 15 or 30 bits long. In C180 mode the instructions are 16 or 32 bits long. C180 BDP instruction descriptors have 32 bits. The instruction opcode addresses the First Level Instruction Decoder, which issues supplementary information needed for instruction execution. The instruction parcel and decoder output are placed in an expanded format before loading into the Buffer Rank 2, 3, 10, 11, and 12 registers. From the Buffer Rank 12 Register the instruction opcode goes to CST to address the first 128-bit micrand needed to execute the instruction. IF also sends immediate operands (operand fields contained in the instruction), control information, and the program address (P lower) of the instruction to ICP.

### Rank Designators

Rank numbers in IF and ICP designate how many major and minor cycles elapse between an instruction's departure from Instruction Assembly and its arrival at a buffer register. A single-digit rank designator is the number of 16-ns minor cycles. The leftmost digit of a two-digit rank designator is the number of 64-ns major cycles and the rightmost digit is the number of additional minor cycles.

The time interval specified by the rank designator is valid only if an instruction's movement in the pipeline is not suspended and no buffer register ranks are bypassed. When Buffer Rank 2, 3, 10, 11, and 12 Registers are empty, the instruction parcel and First Level Instruction Decoder output are loaded directly into Buffer Rank 12 Register.

Movement in the pipeline may be suspended for several reasons. Once an instruction enters ICP Rank 22, its various components remain there until the micrand sequence is finished. The effect is to block all pipeline movement behind the Rank 22 instruction.

An individual micrand may stall in Rank 22 if an operand the micrand needs is being acted upon by another micrand and shortstop paths cannot resolve the conflict.

A micrand may remain in Rank 32 as long as the functional area it needs to use is busy. Also, if a micrand makes a memory reference and Segment Map Miss activates in SM, a hardware-controlled Segment Descriptor fetch sequence occurs. Microcode control is suspended in the CP and the micrand stalls in Rank 32 until the required Segment Descriptor returns from CM.

A micrand executing in Rank 41 typically does not move to Rank 50 with a valid result until Response returns from the functional area the micrand is using.

### Conditional Branching

A conditional branch instruction, which initiates a new instruction sequence, leaves IF and proceeds through ICP before the associated condition can be tested in ALN. If ALN determines at Time 43 that the condition has not been met, ALN sends ALN Unbranch to IF. ALN Unbranch indicates the RNI sequence must be substituted for the instruction sequence addressed by the branch instruction. Rank 50 P, the preserved address of the branch instruction, enters the Branch Address Adder, where it is added to a constant of four. The new address points to the instruction following the branch instruction in the RNI sequence.

### C170 Mode Address Out of Range (AOR) Detector

The C170 Mode AOR Detector performs C170 range testing for instruction addresses.

#### NOTE

Because the Branch Address Adder does not supply end-around carries, no dedicated AOR test hardware is required to detect adder overflow. If the adder produces a result that exceeds the C170 address length (21 bits), the carry out is to three zero bits to the left of the address. When set, any of these bits causes the detector to sense an AOR condition.

### CST

The CST RAM contains microcode required to execute the C170 and C180 instruction sets. The Disassembly Network, used to disassemble 64-bit data words, also is in CST.

### Microcode

The CST RAM contains the C170 and C180 instruction micrand sequences, consisting of one or more micrands for each instruction. A micrand consists of a 64-bit General Micrand and a 64-bit Functional Unit Micrand. The General Micrand and the i, j, and k designators from ICP govern reading and writing of OPI Register File locations. The General Micrand also controls selection of Immediate Operands in OPI. The Functional Unit Micrand controls operations in ALN, BDP, LM, AC, and SM. The functional unit field within the General Micrand determines which functional area(s) uses the Functional Unit Micrand.

An instruction's Rank 12 Opcode addresses the starting location of a micrand sequence in the CST RAM. The Micrand Sequence Control (MSC) Field in each General Micrand determines the address of the next micrand. If an error or exception occurs while an instruction is executing, ICC typically generates a Microtrap Code at Time 52 to address an interrupt microcode routine in the CST RAM.

### Disassembly Network

The CST Disassembly Network disassembles 64-bit Register File Read Data or Functional Unit Micrand Register data into 8-bit bytes for transfer to MAC. The CST Disassembly Network obtains General Micrand or Functional Unit Micrand data via the Functional Unit Micrand Register. The data proceeds to MAC on the MAC Data bus. PFS Bus Data bytes also can enter the MAC Data bus through this network.

### ICP

ICP provides five buffer ranks, corresponding to five stages in the execution of a micrand.

### Rank 13 - Micrand Access

Program Address (P) and the Instruction Mux Bits leave IF's Buffer Rank 12 Register for ICP. At the same time, the Rank 12 Opcode goes to Microcode Address Control in CST to select a micrand. P and the Instruction Mux Bits load into the Rank 13 Register.

### Rank 22 - Operand Selection

When Rank 13 j, k Operand Select, Immediate Operand, P, and other control signals load into the Rank 22 Register at Time 22, the General Micrand loads into the General Micrand Register in CST. The Rank 22 Register remains latched until the instruction exits after the execution of its micrand sequence.

From Time 23 to Time 31, the General Micrand's RDSa, b, and c fields control addressing of the Register File or selecting of Immediate Operands. Concurrently, i, j, j+1, k, and k+1 designators from the Rank 22 Register enter Register File Address Select and Rank 22 P and Immediate Operand enter OPI's Immediate Operand Select network. (j+1 and k+1 designators are used in floating-point arithmetic operations.) The selected operands are then sent to SM, AC, or ALN.

### Rank 32 - Functional Area Selection

At Time 32, the RDS Write (RDSw) field (RDSa or RDSd) enters a minipipe delay network within OPI to control writing into the Register File at Time 52. RDSw and a portion of the General Micrand from CST load into the Rank 32 Register along with j, k Operand Selects, P, and various control signals from the Rank 22 Register. The RDS Write field later returns to OPI to control writing into the Live Registers. The functional unit field of the General Micrand generates the appropriate Go signal after the preceding micrand's Response returns to ICP. Go selects a functional area to perform the task specified by the Functional Unit Micrand. Go usually is issued if the selected functional unit is not busy.



#### Rank 41 - Functional Unit Micrand Execution

Once Go activates, the remaining General Micrand fields and instruction information load into the ICP Rank 41 Register. Typically, they remain in the Rank 41 Register until Response returns to ICP from the functional area selected by the micrand. Response permits the next General Micrand field (in Rank 32) to issue a Go signal and transfer itself to Rank 41. The length of time a micrand remains in the Rank 41 Register may exceed one major cycle. The stay normally depends on how long the selected functional area takes to complete its operation and submit Response.

#### Rank 50 - Result Selection

The Rank 50 Register sends Rank 50 Control signals to ICC. They are used in conjunction with errors and exceptions generated during micrand execution to determine if an interrupt routine will be activated. The Rank 50 Untranslatable Pointer (UTP) Register saves instruction addresses or operand addresses for examination if a Page Table search without find exception occurs.

#### OPI

OPI contains the Register File and various Live Registers. Registers in the Register File hold operands and exchange package data required for instruction execution. Live Registers supply data for various operations and collect data from the CP operating environment.

#### Register File

The 64-word Register File contains operating registers (A, B and X) for C170 and C180 instructions. It also holds other exchange package information and provides holding registers for intermediate results.

The RDSa, b and c fields from the General Micrand and the RDSd field from the Functional Unit Micrand determine Register File read and write addresses. An RDS field may contain the address, or a code within the field may select an instruction's i, j, j+1, k, or k+1 designator as the address. Separate i designators are available for C170 and C180 mode operations. A Start and X Start register bits become the Register File address in the event a C180 Load or Store Multiple (80, 81) instruction, a call, a return, or C170 exchange jump operation is taking place.

The RDS fields also can select an Immediate Operand, P, or Live Register Read Data instead of Register File Read Data for loading into the Operand Registers.

#### DAI Register

The DAI Register can select result data from ALN, LM, or AC for input to the Register File or the Live Registers. Result data is available at the same time ICP Rank 50 is loaded. The DAI Register also can select the Functional Unit Micrand for input to the Register File.

### Live Registers

The Live Registers in OPI contain control information for various C170 and C180 CP operations. Some of the Live Registers are written only under microcode control. Normally they are loaded from the exchange package at the same time the exchange package enters the Register File from CM.

Some of the write only Live Registers also may be loaded during execution of C180 Copy from Xk per (Xj) instructions (OF).

The CP uses the constant output of these Live Registers during on-line operations. The Live Registers give the CP quicker access to exchange package information, which otherwise would have to be obtained from the Register File.

Other Live Registers in OPI are considered to be read only under microcode control. The process interval and system interval timers are examples. The contents of read only registers change as a consequence of changes in the CP hardware environment. These changes typically impact system operation and require monitoring.

### ICC

When an error or exception occurs while a micrand is executing, ICC selects and synchronizes the start of an interrupt routine.

Errors and exceptions accumulate in the Monitor and User Condition Registers and are examined in the Condition to Mask Comparators. C170 mode error exit conditions load into the Exit Mode Condition Register for processing in Exchange Interrupt Control. The Monitor, User, and Exit Mode Mask Register Bits enable ICC to interrupt selectively. Those errors and exceptions in the Monitor and User Condition Registers which are stackable (deferred interrupts permitted) trigger interrupts only when the corresponding mask bits are set. Mask bits determine the type of mandatory interrupt that will occur for unstackable errors and exceptions. Exit Mode Mask Register Bits enable C170 exchanges when set.

### Microtrap Code

The Halt, Exchange, Trap Interrupt Control and Retry Control produces Microtrap Code. The code, which determines the type of interrupt, is based on control signals supplied by ICC and the error inputs. The code addresses the appropriate microcode routine in the CST RAM.

### Clear Pipe

Clear Pipe becomes active if ALN discovers that the required condition is not met during execution of a conditional branch instruction. Instructions that have been requested by the branch instruction must be removed by the pipeline. Clear Pipe also activates to empty the pipeline of instructions before executing an interrupt routine.

### ALN

ALN contains circuitry for integer and floating-point arithmetic, operand shifting, logical operations, normalizing of floating-point operands, and conditional branch comparisons.

The Normalize Encoder performs all normalize operations. The Multiply/Divide Network carries out multiplies and divides of integers and floating-point coefficients. The Shifter Ranks 1 and 2 and the Shift Count Generator act together to perform shift operations. The 97-bit Large Adder performs addition, subtraction, and logical operations on integers and floating-point coefficients. The Exponent Arithmetic Network does exponent calculations. Branch Condition Control evaluates operands to determine if a conditional branch instruction should proceed.

The B and C Operands from OPI are the principal data inputs to ALN. ALN Result returns to OPI for entry into the Register File. Other data inputs are BDP Convert Binary and the AC Shift Count, a selectable input which typically controls the amount and direction of a shift operation. BDP Convert Binary is a byte input which a multiply algorithm assists in converting from decimal to binary form.

ALN generally performs one's complement arithmetic in C170 mode and two's complement arithmetic in C180 mode.

Go initiates ALN arithmetic operations, which are controlled by the Functional Unit Micrand.

#### AC

AC contains the hardware to create CM byte addresses. AC also supplies data to and receives results from BDP during C180 BDP operations.

#### Address Formation

The Address Formation network in AC forms the byte number portion of the address sent to LM for Register File data or BDP stream data. The A/B Operand input to the Address Formation network can contain an operand or address selected by an RDSa or b field in OPI. For BDP instructions the C Operand is an A or B stream length field. Address Offset, normally an Immediate Operand from the instruction, is a third address component. In the C170 mode, the Address Formation network performs 18-bit and 21-bit one's complement addition or subtraction. (Operands are 18-bit quantities and RAC is 21 bits.) In the C180 mode, the Address Formation network performs 32-bit two's complement addition.

The byte number from the Address Formation network merges with an Active Segment Identifier (ASID) from SM or from the A/C, B Stream ASID Register. Together they comprise AC Address, which is a system virtual address (SVA).

#### BDP Instructions

BDP uses CM data provided by LM via AC. AC's data inputs and outputs for BDP operations are LM Read and Write Data. LM Read Data is a 64-bit word disassembled by the A Stream Disassembly network or the B Stream Disassembly network into A Stream Data or B Stream Data bytes. C Stream Data bytes produced in BDP are assembled into a 64-bit word in the C Stream Assembly network and sent to LM as LM Write Data. OPI can also send C Operand data to LM as LM Write Data.

Since the A and C streams are never busy at the same time, they share common hardware within AC - the A/C Stream ASID Register, the A/C Stream Address and Length counters, and the A/C Stream Disassembly/Assembly Network.

### Load/Store Instructions

Load Byte and Load Bit instructions, which transfer data from any location within a memory word to the least significant byte or bit of an operand in the Register File, are processed in AC. The A/C Stream Disassembly/Assembly network shifts the LM Read Data byte or bit and sends it to OPI as Load Data. For Store Byte and Store Bit instructions, the A/C Stream Disassembly/Assembly network shifts Register File data from the least significant position to the desired location in a memory word.

#### NOTE

Because CM does not directly support bit addressing, store bit operations are performed by means of read and set lock and read and clear lock functions for one and zero states, respectively. These functions impose additional requirements on the AC data path to LM.

AC also processes C180 Load and Store Word, Address, and Multiple Word instructions to transfer 64 bits between memory and the Register File.

### AC Control

A Functional Unit Micrand field accompanied by Go provides primary control of AC operations. Soft control memories, loaded from the MAC Data bus during system initialization, contain control signals that are similar to micrand bits but which are issued each minor clock cycle.

The ALN Shift Count Register holds a shift count for use in ALN during ALN shift operations. ALN shift counts are formed by the same adder within the AC Address Formation Network otherwise used to perform memory address arithmetic.

The C170 Mode Range Tester is similar to the C170 Mode AOR Detector in IF except that it detects operand AOR conditions rather than instruction AOR conditions.

#### NOTE

The C170 Mode Range Tester evaluates addresses produced in the Address Formation Network adder, which has an end-around carry network. If overflow occurs, the adder result may erroneously appear to be in range due to the end-around carry. The C170 Mode Range Tester must sense overflow separately when evaluating a CM address.

## BDP

BDP executes C180 mode business data processing instructions in conjunction with AC. AC links BDP to the rest of the CP. AC supplies A and B Stream Data to BDP from LM and receives C Stream Data results for return to LM.

## Data Flow

BDP works with bytes only and processes no more than 256 byte pairs in a single operation. The operand fields enter BDP through A and B Stream Stages 1 through 4 and reach the ALU at the maximum rate of one byte pair per minor cycle. From the ALU, results pass through Common Stages 5 and 6 and enter the Buffer RAM. Once all result data has entered the Buffer RAM, C Stream Data returns to AC through C Stream Stages 1 through 5. When required, the Binary/Decimal Converter changes bytes leaving C Stream Stage 5 from binary to decimal form and sends the data back to the Buffer RAM by way of the A Stream and the ALU.

Register Files A and B, both with 256-byte capacity, are available to process Compare Collate and other instructions for which tabular information is required. When placed in a Register File, that information is used to control processing of data. An Edit instruction requires that a source and a mask field be entered into the Register Files. The fields proceed to the Edit circuitry and produce a result which is stored in the Buffer RAM.

During system initialization, MAC loads RAMs in BDP Control with instruction/data validation data, EBCDIC convert data, and binary/decimal convert data via the MAC Data bus.

## Stream Pausing

Pause A and B Stream signals synchronize the input of A and B Stream Data bytes to the BDP ALU. Pause C stream delays the output of the BDP result until AC forms an address for storing the data in memory. A Functional Unit Micrand controls a BDP operation after Go arrives.

## SM AND LM

SM and LM constitute the virtual memory address circuitry. SM converts the process virtual address (PVA) to an SVA by changing the user's segment number into an ASID. LM converts the SVA to a real memory address (RMA). All memory protection is based on segmentation, with validation performed by SM.

## Segment Map RAM

The Segment Map RAM contains up to 32 most-recently-used Segment Descriptor entries from the process segment table in CM. When the ASID contained in an instruction sequence's Segment Descriptor is latched into the P Descriptor Save Register, it joins IF Address (a byte address) in becoming the IF SVA. Subsequent instruction requests within the segment continue to use the output of the P Descriptor Save Register as the IF ASID.

The contents of the P Descriptor Save Register also serve as the ASID portion of the AC Address (operand SVA) during C180 Load Bytes, Displaced Relative (86) instructions and during all C170 CM operand references. The ASID for other operand SVAs enters AC directly from the Segment Map RAM. Operand and instruction SVAs become RMAs in LM.

If the required Segment Descriptor entry is not available in the Segment Map RAM, a Segment Descriptor Fetch sequence obtains it from the process segment table in CM. Segment Descriptor Fetches occur under hardware control rather than microcode control. Microcode control is suspended during Segment Descriptor fetch memory references.

#### Access Validation Testing

SM performs all of the access validations for addressing memory. Security ring tests are performed, as are key/lock tests and read, write and execute privilege validity tests. The tests are comparisons between Access Control Fields in the Segment Descriptor and the Ring and Keys contained in a PVA, supplied by the accessor. SM Access Violation is an exception indicating that an illegal memory reference has been attempted. SM Access Violation initiates an interrupt routine by producing a Microtrap Code in ICC. A Functional Unit Micrand field controls various SM operations.

#### Cache Memory

LM contains up to 4K words of high-speed Cache Memory to provide storage for the most recently used CM data. A four-word block of CM Read Data words enters Cache Memory, a word at a time in an order that is not necessarily predetermined. The word of interest in four-word block goes to other destinations within the CP as LM Read Data. When the CP addresses the data a second time, the data is recalled from Cache Memory as LM Read Data. An intermediate storage device, Cache Memory enhances CP performance in reading CM data.

When the CP sends LM Write Data to CM, the CP writes the Cache Memory location containing the same CM data the write is modifying in CM.

#### Page Map RAM

LM contains 128 entries from the CM system page table (SPT) in the Page Map RAM. An SPT entry is required to convert an SVA into an RMA to access CM when the requested data is not available in Cache Memory. The presence of the most recently used SPT entries in the Page Map RAM accelerates addressing of CM. The Map Address from the SVA obtains the Page Frame Address (PFA). The PFA combines with the Page Offset (PO) from the SVA to generate the RMA.

If the desired SPT entry for a memory access is not in the Page Map RAM, Page Table Search Control creates Page Table Address (PTA) Hash. PTA Hash is an RMA into the SPT created from the SVA's ASID and page number. The required SPT entry returns from CM as CM Read Data. It enters the Page Map RAM at the same time it is gated back to CMC as the PFA portion of a new RMA. The CM Read Data words addressed by this second RMA enter Cache Memory at the same time the word of interest goes to a functional area as LM Read Data.

#### CMC Tag

During CM read operations the CP typically uses CMC Tag to determine where to send data entering the CP from CM. The RMA Network and Control generates the tag to accompany an RMA to CMC. The tag enters a delay circuit in CMC which synchronizes it with the data the RMA obtains from CM. CMC Tag returns from CMC to LM Control to guide LM Read Data into the CP and, when applicable, Cache Memory. During an exchange jump, LM Control sends CMC Tag to OPI to enter LM Read Data into the Register File and, in some cases, the Live Registers.

A/C Stream LM Requests, IF Request, or Instruction Issue Request Enable initiate LM operations. Functional Unit Micrand fields control the operations and Stream A/C, B Response, IF Response, or Instruction Issue Response signals indicate their completion.

#### CMC AND CM

LM (A), LM (B), IOU and an auxiliary port all have access to CM through CMC. CMC contains Conflict and Bank Busy Control, which resolves bank conflicts and simultaneous CM request conflicts. Memory is contained in a single stand-alone cabinet, available to CP-0 and an optional CP-1.

#### Refresh

The Conflict and Bank Busy Control generates a refresh request every 15 clock periods and sends it to Bank Control via the Go Bank signal. To refresh memory, a read cycle is performed on each row address of each memory array. When refresh is not active, Conflict and Bank Busy Control prioritizes and initiates other requests such as write, read, and partial write.

#### Write

During a CM write operation, Address, Control, and Write Data enter CMC through an Input Port. The Input Port routes the Bank Code, part of the Address, to the Conflict and Bank Busy Control. There the Bank Code produces a Go Bank signal, which passes through Bank Control and selects one of eight Memory Banks in CM. The remaining Address and Control signals also proceed to CM's Bank Control. Write Data enters the Write Error Correction Code (ECC) Generator, where an ECC is formed to accompany the data to Bank Control. Go Bank starts the write sequence in the Memory Bank, ultimately resulting in data being stored in that bank.

#### Read

During a CM read operation, Bank Code, Address, and Control signals become active as they do for a write, except that a write request is absent. Read Data departs a Memory Bank along the bidirectional read/write path, passes through Bank Control, and enters the Single Error Correction, Double Error Detection (SECDED) network in CMC. SECDED corrects any single bit errors and flags double bit errors. An Output Port then sends the Read Data to the device that initiated the read request.

#### Partial Write

CMC is capable of modifying an individual byte or bytes within a memory word. The operation requires two passes. The modifying write data enters CMC through an Input Port, goes through the Write ECC Generator and loads into a holding register in CM's Bank Control. Write Data from that holding register and Read Data from a Memory Bank (by way of Bank Control and SECDED) enter the Partial Write (PW) Network and ECC Generator in CMC. The combination of the two words creates a PW Data word, which returns to memory after an ECC is generated. Mark bits control the Partial Write Network. Memory word bytes which have corresponding mark bits set are modified. The remaining bytes return to CM unchanged.

#### Maintenance Registers

The Maintenance Registers monitor CMC and CM errors. The Maintenance Registers also provide

the means for testing and reconfiguring CM for maintenance purposes. MAC can transfer CMC Data error signals from a Maintenance Register to the MCU. MAC also loads various Maintenance Registers with MAC Data.

#### INSTRUCTION EXECUTION

An exchange sequence starts normal execution of program instructions. All instructions have the following common steps.

1. IF Instruction Assembly parcels out instruction.
2. Opcode addresses First Level Instruction Decoder.
3. Decoder output and instruction parcel load into IF buffer registers.
4. Rank 12 Opcode addresses first word of instruction's micrand sequence in CST.

Rank 12 P and Instruction Mux Bits load into ICP Rank 13 Register.

5. General Micrand loads into General Micrand Register in CST.

Contents of instruction pipeline load into Rank 22 Register.

The i, j, and k fields from ICP, RDSa, b, and c fields from General Micrand, and RDSd field from Functional Unit Micrand control reading and writing in Register File. Pipeline data remains latched in Rank 22 Register, feeding Rank 32 Register, until instruction exits.

6. General Micrand and pipeline data load into Rank 32 Register. Go activates functional area specified by General Micrand Functional Unit field.

From this point, instruction execution can proceed along a number of paths and depends on the unique requirements of each instruction specified by its micrand sequence. The C180 instruction descriptions that follow demonstrate various uses of system hardware.

#### INTEGER SUM, (Xk) REPLACED BY (Xk) PLUS (Xj)

The arithmetic operation performed by this instruction takes place in ALN with operands provided from the Xk and Xj registers in the Register File. The sum returns to the Xk Register.

1. General Micrand RDSb and c fields gate Rank 22 j and k Operand Select through OPI's Register File Address Select to Register File.
2. Functional Unit Micrand departs CST at Time 31. It determines that add operation will be performed in ALN's Large Adder.
3. ICP generates Go at Time 32.



4. Operands addressed by j and k designators in Register File proceed to ALN at Time 40 as B and C Operands. C Operand goes to Large Adder. B Operand's path to Large Adder is through Multiply/Divide Network (no op). Response goes to ICP.
5. Large Adder Result enters ALN Output Mux. At Time 50 ALN Result is available to OPI.
6. Register File Address (k designator) arrives at Register File from delay circuit in Register File Address Select.

ALN Result passes through DAI Register and enters Register File Xk register at Time 52.

If at any time before new data is written into the Xk register an error or exception occurs, ICC evaluates the error or exception to determine if an interrupt routine is required. The interrupt clears the pipeline, blocks the write into the Register File, and proceeds with a different micrand sequence.

#### LOAD Xk FROM (Aj) DISPLACED BY Q

The data word stored in CM location (Aj) plus eight times Q loads into the Register File Xk register.

This instruction uses the Address Formation network in AC and the PVA-to-RMA conversion circuitry in SM and LM to address a CM location.

The instruction sequence is as follows:

1. General Micrand's RDSa field selects Rank 22 j Operand Select in OPI to become A register read address.

RDSd field selects Rank 22 k Operand Select as X register write address.

Designator k enters delay circuit in Register File Address Select to address Register File when requested CM data arrives from LM.

Rank 22 Immediate Operand provides quantity Q to Immediate Operand Select in OPI, where it is multiplied by eight (left-shifted three places).

2. Eight times Q goes to Address Formation network in AC as Address Offset.

Byte number portion of PVA from Aj register enters Address Formation network as A/B Operand.

PVA segment and ring numbers from Aj register enter PVA Register in SM as Register File Data.

3. AC Address Formation network adds Address Offset and A/B Operand to form Address Adder Byte Number, which becomes AC Address (lower). Tests determine if Address Specification Error has occurred.

Segment number addresses Segment Map RAM. RAM supplies ASID portion of Segment Descriptor to AC.

PVA ring number goes to Key/Lock, Ring Validity Tests in SM. There Keys from P Register and ring number are compared to Access Control Fields of Segment Descriptor. Tests determine if the segment is invalid or SM Access Violation has occurred.

Functional unit field of General Micrand in ICP issues Instruction Issue Request Enable to LM.

4. ASID and Address Adder Byte Number enter LM as AC Address, the operand SVA.

Instruction transfers to ICP Rank 41. Further movement in pipeline or along write address delay path in OPI is blocked until Response arrives in ICP from LM.

5. When Instruction Issue Request Enable arrives in LM, it initiates Cache Memory Addressing. If Cache Memory contains word, LM sends LM Read Data to DAI Register in OPI and generates Response.

LM simultaneously forms RMA to obtain word from CM, provided SPT entry needed to produce RMA is in LM's Page Map RAM.

6. If word is not present in Cache Memory, LM Address and additional Control signals enter CMC through Input Port and perform CM read operation.
7. CM Read Data returns to LM, where it enters Cache Memory and proceeds to OPI as LM Read Data. LM blocks generation of Response until read data returns from CM. Response is further delayed if SPT entry needed to produce RMA is not in LM's Page Map RAM.
8. Once Response reaches ICP, instruction advances in pipeline and Register File write address in OPI places incoming LM Read Data in Xk register.

DECIMAL SUM, D(Ak) REPLACED BY D(Ak) PLUS D(Aj)

This BDP instruction adds decimal data field D(Ak) to decimal field D(Aj) and returns the result to CM as decimal data field D(Ak).

Ten micrands are required to execute the instruction, which processes data field D(Aj) as A Stream Data, data field D(Ak) as B Stream Data and the result field D(Ak) as C Stream Data. Two 32-bit BDP descriptors accompany the instruction through IF and ICP to specify the field lengths, data types, and address offsets for the A and B streams.

The instruction sequence is as follows:

1. First micrand issues Go to BDP.

Functional Unit Micrand enters BDP and remains latched there through course of instruction to control arithmetic operations.

First two micrands use j and k designators from ICP to address A registers.

Contents of Aj and Ak registers load into scratch registers in Register File.

The next four micrands utilize the contents of the two BDP descriptors to form the addresses needed to read A and B stream data from CM. Two micrands produce the A stream SVA and two micrands produce the B stream SVA.

2. First descriptor loads into Rank 22 Register in ICP. Because j and k designators are now no longer available in instruction pipeline for Register File addressing, subsequent micrands obtain Aj and Ak from scratch registers.
3. Contents for Aj register are read from Register File. Thirty-two least significant bits enter Address Formation network adder in AC as A/B Operand.

Rank 22 Immediate Operand (offset field from first descriptor) reaches Address Formation network as Address Offset.

Third addend is C Operand (A stream byte length field from descriptor).

A stream byte length field also enters A/C Stream Length Counter to decrement when A stream data begins arriving from LM.

Sixteen most significant bits from Aj register (ring and segment number) enter PVA Register in SM as Register File Data.

4. Invalid segment and access violation tests are performed and ASID is read from Segment Map RAM.

Address Offset, A/B Operand, C Operand and -1 input are added in Address Formation network to form rightmost stream address. Sum enters A/C Stream Address Counter. (Output of counter is byte number portion of A stream SVA. Counter addresses successive A stream words in LM.)

ASID proceeds to A/C Stream ASID Register in AC, where together with A/C Stream Address Count, ASID forms AC Address.

AC generates A/C Stream LM Request to start A stream data transfer from LM.

5. Second descriptor loads into Rank 22 Register in ICP and B stream operations begin. B stream address formation circuitry includes adder, B Stream Length Counter, B Stream Address Counter, and B Stream ASID Register. Together they generate and update addresses needed to complete B stream data transfer to AC.
  6. Once addresses are formed and requests submitted, LM returns 64-bit LM Read Data and Stream A/C or B Response to AC. Data enters A/C or B Stream Disassembly network, depending on which response signal is active. A/C or B Stream LM Requests return to LM as required, accompanied by new AC Addresses from counters. Appropriate length counter decrements with each word transfer.
- Disassembled A Stream Data and B Stream Data bytes enter BDP's A and B Stream Stages 1 through 3. Two streams are synchronized in Stage 3 by Pause A and B Stream signals, which return to AC to regulate data flow.
- ALU adds A and B Stream Data one digit at a time and places result in Buffer RAM for future output as C Stream Data.
7. A/C and B Stream LM Requests terminate once length counters in AC reach zero.
  8. AC blocks returning C Stream Data with Pause C Stream while last group of micrands validates C stream PVA and generates SVA. AC Address for C Stream is recalculated B Stream SVA. AC Address is produced from saved contents of Ak Register and second descriptor's length and offset fields.

9. Once C stream SVA enters A/C Stream ASID Register and A/C Stream Address Counter, AC drops Pause C Stream.

AC begins to assemble 64-bit LM Write Data words in C Stream Assembly network.

When 64-bit word is assembled, AC issues A/C Stream LM Request to initiate write sequence to LM.

10. LM Write Data enters CM and associated Cache Memory locations.

A/C Stream Address Counter forms next memory address. A/C Stream Length Counter decrements. AC continues to submit A/C Stream LM Requests and AC Addresses for each assembled word until A/C Stream Length Counter equals zero. LM acknowledges receipt of data with Stream A/C Response.

## SECTION 2

### MAINTENANCE CHANNEL/CP INTERFACE



---

MAINTENANCE ACCESS CONTROL (MAC)

The following text is to be used in conjunction with MAC Level 1 and Level 3 diagrams for an explanation of MAC operation.

MAC performs the following operations:

- Reads and writes CP-resident registers and memories.
- Monitors and records error information.
- Reconfigures hardware.
- Verifies error detection and correction hardware.

These operations are controlled by a dedicated peripheral processor in the IOU, programmed to act as the MCU.

FUNCTION CODE

MAC responds to a Function Code sent by the MCU on the Maintenance Channel. The Function Code indicates the operation to be performed and the affected register. Table 2-1 lists the CP-resident registers to which the MCU has access.

Table 2-1. Registers Addressable by Copy State Instruction, MCU (Sheet 1 of 2)

LOC	TYPE CODE	REGISTER ADDRESS	REGISTER NAME	IOU READ	IOU WRITE	CP READ	CP WRITE	SIZE (BITS)
MAC	0	00	STATUS SUMMARY	DIRECT	NO ACCESS	NO ACCESS	NO ACCESS	6
	↑	10	ELEMENT ID	MAC	NO ACCESS	UNPRIVILEGED	NO ACCESS	32
		11	PROCESSOR ID	MAC	NO ACCESS	UNPRIVILEGED	NO ACCESS	1
		12	OPTIONS INSTALLED	MAC READ	NO ACCESS	UNPRIVILEGED	NO ACCESS	4
RF		13	VIRTUAL MACHINE CAPABILITY LIST	MICROCODE	NO ACCESS	UNPRIVILEGED	NO ACCESS	1
PMF		21	PMF KEYPOINT DATA	MAC	CLEAR ONLY	NO ACCESS	NO ACCESS	16 X 64
		22	PMF CONTROL REGISTER 1	MAC	MAC	NO ACCESS	NO ACCESS	64
		22	PMF CONTROL REGISTER 2	MAC	MAC	NO ACCESS	NO ACCESS	64
		22	PMF COUNTERS A0 & B0	MAC	MAC	NO ACCESS	NO ACCESS	64
		22	PMF COUNTERS A1 & B1	MAC	MAC	NO ACCESS	NO ACCESS	64
		22	PMF COUNTERS A2 & B2	MAC	MAC	NO ACCESS	NO ACCESS	64
		22	PMF COUNTERS A3 & B3	MAC	MAC	NO ACCESS	NO ACCESS	64
MAC		30	DEPENDENT ENVIRONMENT CONTROL	MAC	MAC	UNPRIVILEGED	NO ACCESS	32
CST		31	CONTROL MEMORY ADDRESS	DIRECT	DIRECT	UNPRIVILEGED	NO ACCESS	10
		32	CONTROL MEMORY BREAKPOINT	DIRECT	DIRECT	UNPRIVILEGED	NO ACCESS	10
IF,SM		40	P REGISTER	MICROCODE	MICROCODE	UNPRIVILEGED	NO ACCESS	64
RF		41	MONITOR PROCESS STATE POINTER	MICROCODE	MICROCODE	UNPRIVILEGED	NO ACCESS	16
ICC		42	MONITOR CONDITION REGISTER	MICROCODE	NO ACCESS	UNPRIVILEGED	NO ACCESS	16
		43	USER CONDITION REGISTER	MICROCODE	MICROCODE	UNPRIVILEGED	NO ACCESS	16
ICP		44	UNTRANSLATABLE POINTER	MICROCODE	MICROCODE	UNPRIVILEGED	NO ACCESS	48
OPI		45	SEGMENT TABLE LENGTH	MICROCODE	MICROCODE	UNPRIVILEGED	NO ACCESS	32
		46	SEGMENT TABLE ADDRESS	MICROCODE	MICROCODE	UNPRIVILEGED	NO ACCESS	32
RF		47	BASE CONSTANT	MICROCODE	MICROCODE	UNPRIVILEGED	NO ACCESS	32
LM		48	PAGE TABLE ADDRESS	MICROCODE	MICROCODE	UNPRIVILEGED	NO ACCESS	32
		49	PAGE TABLE LENGTH	MICROCODE	MICROCODE	UNPRIVILEGED	NO ACCESS	8
OPI		4A	PAGE SIZE MASK	MICROCODE	MICROCODE	UNPRIVILEGED	NO ACCESS	7
RF		50	MODEL DEPENDENT FLAGS	MICROCODE	MICROCODE	UNPRIVILEGED	NO ACCESS	16
ICC		60	MONITOR MASK REGISTER	MICROCODE	MICROCODE	UNPRIVILEGED	MONITOR	16
RF		61	JOB PROCESS STATE POINTER	MICROCODE	MICROCODE	UNPRIVILEGED	MONITOR	32
OPI	↓	62	SYSTEM INTERVAL TIMER	MICROCODE	MICROCODE	UNPRIVILEGED	MONITOR	32
	0	80-89	PROCESS FAULT STATUS	MAC	MAC	UNPRIVILEGED	GLOBAL	400



Table 2-1. Registers Addressable by Copy State Instruction, MCU (Sheet 2 of 2)

LOC	TYPE CODE	REGISTER ADDRESS	REGISTER NAME	IOU READ	IOU WRITE	CP READ	CP WRITE	SIZE (BITS)
MAC	0	A0	PROCESSOR TEST MODE	MAC	MAC	UNPRIVILEGED	GLOBAL	48
OPI	↑	C0-C3	TRAP ENABLES	MICROCODE	MICROCODE	UNPRIVILEGED	LOCAL	8
RF		C4	TRAP POINTER	MICROCODE	MICROCODE	UNPRIVILEGED	LOCAL	48
		C5	DEBUG LIST POINTER	MICROCODE	MICROCODE	UNPRIVILEGED	LOCAL	48
		C6	KEYPOINT MASK	MICROCODE	MICROCODE	UNPRIVILEGED	LOCAL	16
		C7	KEYPOINT CODE	MICROCODE	MICROCODE	UNPRIVILEGED	LOCAL	32
		C8	KEYPOINT CLASS NUMBER	MICROCODE	MICROCODE	UNPRIVILEGED	LOCAL	8
OPI		C9	PROCESS INTERVAL TIMER	MICROCODE	MICROCODE	UNPRIVILEGED	LOCAL	32
RF		E0-E1	CRITICAL FRAME FLAG	MICROCODE	MICROCODE	UNPRIVILEGED	UNPRIVILEGED	1
		E2-E3	ON CONDITION FLAG	MICROCODE	MICROCODE	UNPRIVILEGED	UNPRIVILEGED	1
		E4	DEBUG INDEX	MICROCODE	MICROCODE	UNPRIVILEGED	UNPRIVILEGED	8
OPI	↓	E5	DEBUG MASK	MICROCODE	MICROCODE	UNPRIVILEGED	UNPRIVILEGED	8
ICC	0	E6	USER MASK	MICROCODE	MICROCODE	UNPRIVILEGED	UNPRIVILEGED	16
CMC	A	00	STATUS SUMMARY	DIRECT	NO ACCESS	NO ACCESS	NO ACCESS	8
	↑	10	ELEMENT ID	DIRECT	NO ACCESS	NO ACCESS	NO ACCESS	32
		12	OPTIONS INSTALLED	DIRECT	NO ACCESS	NO ACCESS	NO ACCESS	24
		20	ENVIRONMENT CONTROL	DIRECT	DIRECT	NO ACCESS	NO ACCESS	16
		21	BOUNDS REGISTER	DIRECT	DIRECT	NO ACCESS	NO ACCESS	36
		A0	CORRECTED ERROR LOG	DIRECT	DIRECT	NO ACCESS	NO ACCESS	40
		A4	UNCORRECTED ERROR LOG 1	DIRECT	DIRECT	NO ACCESS	NO ACCESS	64
	↓	A8	UNCORRECTED ERROR LOG 2	DIRECT	DIRECT	UNPRIVILEGED	NO ACCESS	50
	A	B0	FREE RUNNING COUNTER	NO ACCESS	DIRECT	UNPRIVILEGED	NO ACCESS	48
CST	1		CONTROL STORE MICROCODE	DIRECT	DIRECT	NO ACCESS	NO ACCESS	
MAC	3		MAC REF ROM	SWEEP	DIRECT	NO ACCESS	NO ACCESS	
AC,ALN	4		SOFT CONTROL MEMORY	SWEEP	DIRECT	NO ACCESS	NO ACCESS	
BDP	5		BDP RAMS	SWEEP	DIRECT	NO ACCESS	NO ACCESS	
IF	6		INSTRUCTION DECODE RAMS	SWEEP	DIRECT	NO ACCESS	NO ACCESS	
OPI	7		REGISTER FILE	DIRECT	DIRECT	NO ACCESS	NO ACCESS	

The Function Code has the following format.

0	1	2	3	4	5	6	7	8
OPCODE			TYPE CODE			PARITY		

Opcode and Type Code translations are:

<u>Opcode</u>	<u>Type Code</u>
0 - Stop	0 - P3 Process Register
1 - Start	1 - CST RAM
4 - Read	3 - MAC Reference ROM
5 - Write	4 - Soft Control Memories
6 - Master Clear	5 - BDF Control Memories
7 - Clear Error	6 - IF First Level Instruction Decoder
8 - Echo	7 - Register File
2,3,9-F - Unused	A - CMC Maintenance Registers
	2,8,9,B-F - Unused

The Type Code indicates the memory device or register in which the designated operation is to be performed. Types 0, 3 through 7 and A operations require an address input. The IOU sends an address to MAC via the Maintenance Channel after IOU sends the Function Code. Table 2-1 lists specific addresses for Type 0 and A operations. Addresses for Type 3 through 7 operations are not fixed and can point to any location in the associated memory.

#### MAINTENANCE CHANNEL PROTOCOL

Data transfers between MAC and the IOU on the Maintenance Channel are controlled by channel protocol.

All control signals arriving in MAC are 50 ns long. Signals transmitted from MAC to the IOU are 64 ns long. These signals are asynchronous, with resynchronization occurring in the receiving unit.

#### Function

Function (MAC 1.0) indicates an eight-bit Function Code is available to MAC as a Maintenance Channel Data input.

#### Ready Out

Ready Out (MAC 1.0) indicates an eight-bit byte (other than Function Code) is available to MAC as Maintenance Channel Data or that the IOU is ready to receive a Maintenance Channel Data Output byte from Data Mux 3 (MAC 1.0).

#### Activate Out

Activate Out (MAC 1.0) indicates to MAC that data bytes (other than Function Code) will arrive as Maintenance Channel Data accompanied by Ready Out. Alternatively, Activate Out indicates the IOU is ready to accept bytes (other than Function Code) from MAC on the Maintenance Channel Output bus (MAC 1.0).

#### Disconnect Out

IOU generates Disconnect Out to indicate IOU has disconnected from the Maintenance Channel. PMF employs the signal; MAC does not.

#### Ready In To IOU

MAC generates Ready In To IOU (MAC 1.0) to accompany data bytes from MAC to the IOU or after receiving a data byte (excluding Function Code) from the IOU.

#### Disconnect In To IOU

MAC sends Disconnect In To IOU (MAC 1.0) after receiving Function and associated Function Code from IOU.

#### Error In To IOU

MAC generates Error In To IOU (MAC 1.0) if MAC detects a Maintenance Channel parity error or illegal Function Code.

### STATUS SIGNALS

#### Status Summary

Status Summary (MAC 1.0) is a static input from MAC to IOU, reflecting any of several conditions sensed in the CP or CMC.

#### Exchange Accept

MAC sends Exchange Accept (MAC 1.0) to the IOU when the CP completes a C170 Exchange Jump operation.

### PROCESSOR FAULT STATUS (PFS)

Fault isolation information from all functional areas of the CP except CMC and CM is recorded in the PFS Register (MAC 1.0). There are 400 locations available for error signal storage. Bit assignments appear in Maintenance Aids section of Maintenance and Parts Data manual.

### DEPENDENT ENVIRONMENT CONTROL (DEC)

The DEC Register controls the internal configuration and operation of the CP. Bit assignments appear in Maintenance Aids section of Maintenance and Parts Data manual.

## PROCESSOR TEST MODE (PTM)

The PTM Register causes selected parity checkers in the CP to detect parity errors and verifies the errors are properly reported and recorded. Bit assignments appear in Maintenance Aids section of Maintenance and Parts Data manual.

## MAC OPERATIONS

The four major categories of MAC operations are:

- IOU immediate operations.
- IOU direct reads and writes.
- IOU microcode-assisted reads and writes.
- CP reads and writes.

Immediate operations clear CP or CM maintenance registers and stop or start the CP.

Direct operations permit the IOU to read or write various CP-resident registers without microcode assistance from the CP. Data transfer between the maintenance channel and the addressed register is via Data Muxes 1 through 6 (MAC 1.0) and, in some cases, the MAC Data Bus (MAC 1.0). Refer to figure 1-1 for a detailed view of the MAC Data Bus. Addresses for direct operations using the MAC Data Bus are transmitted on the MAC Address Bus (MAC 1.0). Refer to figure 2-1 for a detailed view of the MAC Address Bus. The MAC Byte Select path (MAC 1.0) is used to gate bytes arriving on the MAC Data Bus to proper byte positions within various Control Memory words. Refer to figure 2-1 for a detailed view of the MAC Byte Select Bus.

The MAC Data Bus provides the write data path between MAC Data Mux 6 (MAC 3.0) and various Control Memories, Register File and PFS Registers. The bus serves as a read data path from the PFS Registers and the Register File to MAC Data Mux 5 (MAC 3.0).

When Select PFS Bus is active, the CST Disassembly Network shown in figure 2-1 permits PFS Bus Data bytes to enter the MAC Data Bus directly through Disassembly Mux 2 (CST 3.5). When Select PFS Bus is inactive, bytes enter the data bus from Disassembly Muxes 1 and 2.

Select Register File to Disassembly Network controls entry of two 64-bit data words into the Disassembly Muxes. A word is disassembled into eight 8-bit bytes by CST Byte Count Bits 0 through 2.

The IOU requires microcode assistance to read or write various Type 0 registers. The microcode sequence transfers data between the addressed register in the CP and the Register File (MAC 1.0). Output to the Maintenance Channel is from the Register File through the CST Disassembly Network (MAC 1.0) and Data Muxes 5 and 3. Write data is input to the Register File through the Maintenance Channel Data Register (MAC 1.0), Data Mux 6 and AC Soft Control Memory (MAC 1.0).

The CP reads and writes certain MAC-resident registers in conjunction with C180 Copy State (OE, OF) instructions. The data path for PFS, DEC, or PTM register reads is through the CST Disassembly Network, Data Muxes 5 and 6, and AC Soft Control Memory to the Register File. The PFS or PTM write data path is from the Register File through the CST Disassembly Network, Data Muxes 5 and 6, AC Soft Control Memory and Register File on the MAC Data Bus.



Figure 2-1. MAC Data, Address, Byte Select Buses

## FUNCTION

The Function sequence, first step of all IOU operations in MAC, includes the following events.

1. Function arrives at Maintenance Channel Control Signal Resynchronization Network (MAC 3.3).

Function Code arrives at Maintenance Channel Data Parity Check (MAC 3.0) as Maintenance Channel Data Bits 0 through 7/8.

2. Parity is checked and byte proceeds to Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2).

If parity error occurs, Maintenance Channel Input Parity Error 0 enters Maintenance Channel Control Signal Resynchronization Network translator.

3. Resynchronized Function Out departs 48-ns Delay and Resynchronization Network (MAC 3.3) and resets Control Word 1, 2 and Control Words Loaded FFs.
4. Resynchronized Function Out enters Resynchronization Register (MAC 3.3) and becomes Function 1.
5. Function 1 proceeds to Channel Timing Delay Chain (MAC 3.3). Function 1 also resets Channel Byte Counter (MAC 3.3).

Function 1 resets Partial Address Incrementer (MAC 3.2), and CST Interface FF (MAC 3.4). Function 1 enables clearing of Read and Write FFs (MAC 3.4) at next T1.

Function 1 resets Address Incrementer (MAC 3.6) and activates Byte Transfer or Clear Error Complete and Microcode Operation Complete in Read, Write Control (MAC 3.6).

Function 1 resets MAC Byte Counter (MAC 3.8). Function 1 also activates Clear MAC Busy in Clear MAC Operation Translator if lengthy microcode-assisted read operation is in progress in CP. Clear MAC Busy unhangs MAC.

Function 1 loads Maintenance Channel Opcode and Type Code Registers (MAC 3.2) and Function Code decoding begins.

Function 1 and Maintenance Channel Input Parity Error 0 generate Function Parity Error (MAC 3.3) if parity error has occurred. Function Parity Error enters PFS Translator (MAC 3.8) to generate PFS Register 86 Bit 33 and Error In To IOU. Function Parity Error blocks Function 1 from starting Channel Timing Delay Chain.

6. An illegal Opcode or Type Code in the Function Code activates Error In To IOU in PFS Translator (MAC 3.8) at Channel Timing Chain T4.

## IOU IMMEDIATE OPERATIONS

Of the seven MAC operations, Stop, Start, Master Clear, and Clear Error are defined as Immediate Operations. Immediate Operations require only Function and the accompanying Function Code.

#### STOP (FUNCTION CODE = 0X)

The Stop operation causes the CP to stop at the end of the current instruction and sets Status Summary Bit 1 (Processor Halted).

The operation forces the Instruction Control Pipeline invalid and the CST Micrand sequence into an idle loop until restart occurs.

1. Function and Function Code arrive in MAC. Refer to Function description for details.  
Opcode of zero activates Operation Decode Equals 0 in Opcode Decoder 0 (MAC 3.2).  
Type Code has no significance in Stop operation.
2. Function 2 enters Disconnect Translator (MAC 3.3) to activate Disconnect In To IOU.  
Function 1 starts Channel Timing Delay Chain (MAC 3.3).
3. Channel Timing Chain T2 and Operation Decode Equals 0 generate Stop in CST Start, Stop Control (MAC 3.4).
4. Stop sets Stop FF in Start/Stop Control (CST 3.0).
5. Stop forces Trap Address Bits 0 through 10/Parity to 004<sub>16</sub> in Trap Address Formation Network (CST 3.3) when current instruction exits.
6. Trap Address bits pass through Micrand Address Muxes 2 and 1 (CST 3.4), exiting as Micrand Address MUX 1 Bits 0 through 10/Parity.
7. Micrand Address Mux 1 Bits 0 through 10/Parity enter Micrand Address Mux Register (CST 3.4).
8. CST Address Bits 0 through 10 proceed to CST RAM (CST 3.5), addressing location 004<sub>16</sub>.
9. Micrand sequence at locations 004<sub>16</sub> through 006<sub>16</sub> sets Processor Halted FF (CST 3.0) when the pipeline empties. The micrand sequence loops waiting for Start function from MAC.

#### START (FUNCTION CODE = 1X)

The Start operation permits the CP to begin addressing the CST RAM (CST 3.5) for CP on-line operations.

1. Function and Function Code arrive in MAC. Refer to Function description for details.  
Opcode of one activates Operation Decode Equals 1 in Opcode Decoder 0 (MAC 3.2), clearing MAC Off Line FF (MAC 3.6). Type Code has no significance in Start operation.
2. Function 2 enters Disconnect Translator (MAC 3.3) to generate Disconnect In To IOU.  
Function 1 starts Channel Timing Delay Chain (MAC 3.3).
3. Channel Timing Chain T2 and Operation Decode Equals 1 activate Start in CST Start, Stop Control (MAC 3.4).

4. Start sets CST On Line FF and clears Processor Halted FF in Start/Stop Control (CST 3.0).
5. On Line and inactive Processor Halted enable Advance Micrand to enter Micrand Advance Control Holding Register (CST 3.0).
6. Enable MSC or Interrupt and On Line produce Enable Micrand Address Register in MAC Off Line Control (CST 3.0).
7. Enable Micrand Address Register permits loading of Micrand Address Mux Register (CST 3.4) from source determined by Micrand Address MUX 1, 2 Selects.

#### MASTER CLEAR (FUNCTION CODE = 60 OR 6A)

Master Clear operation master clears either the CP or CM, depending on the Type Code value.

1. Function and Function Code arrive in MAC. Refer to Function description for details.  
 Opcode of six activates Operation Decode Equals 6 in Maintenance Channel Opcode, Type Code Register Decoder (MAC 3.2). Type Decode Equals A activates in Type Code Decoder 1 (MAC 3.2) for CM master clear, or Gated Type Decoder Equals 0 activates in Type Code Decoder 0 (MAC 3.2) to master clear CP.
2. Function 2 enters Disconnect Translator (MAC 3.3) to generate Disconnect In To IOU.  
 Function 1 starts Channel Timing Delay Chain (MAC 3.3).
3. Channel Timing Chain T2 and Operation Decode Equals 6 generate Master Clear (Master Clear FF, MAC 3.4).
4. If Gated Type Decode Equals 0 is active, Master Clear sets MAC Off Line FF (MAC 3.6). MAC Off Line enables loading of various CP Control Memories from IOU via MAC Data Bus.
5. Master Clear FF sets (MAC 3.4). Master Clear or Master Clear CM activates, depending on Type Code value.
6. Channel Timing Chain T4 activates.
7. Operation Decode Equals 6 and Channel Timing Chain T4 generate Soft Control Go Select 1 through 3 in Soft Control Go Decoder (MAC 3.2). MAC addresses location zero (unused address) in all Soft Control Memories during Master Clear operation.
8. Channel Timing Chain T5 clears Master Clear FF.

#### CLEAR ERROR (FUNCTION CODE = 70 OR 7A)

The Clear Error operation clears either the PFS register in MAC or various error status and maintenance registers in CMC, depending on the Type Code value.

1. Function and Function Code arrive in MAC. Refer to Function description for details.



Opcode of seven activates Operation Decode Equals 7 in Opcode Decoder 0 (MAC 3.2).

(If Type Code is A, refer to steps 2 through 6; if Type Code is 0, refer to steps 7 through 20.)

2. Type Code of A activates Type Decode Equals A in Type Code Decoder 1 (MAC 3.2).
3. Function 2 enters Disconnect Translator (MAC 3.3) and generates Disconnect In To IOU.  
Function 1 starts Channel Timing Delay Chain (MAC 3.3).
4. Channel Timing Chain T4, Type Decode Equals A, and Operation Decode Equals 7 generate Clear CMC Error (MAC 3.2).
5. Clear CMC Error clears A0, A4 and A8 registers (CMC 3.13) and Error Status Network FFs (CMC 3.14).
6. Channel Timing Chain T4 becomes inactive, disabling Clear CMC Error.
7. Type Code of zero activates Type Decode Equals 0 and Gated Type Decode Equals 0 in Type Code Decoders 1 and 0 (MAC 3.2), respectively.
8. Operation Decode Equals 7 and Gated Type Decode Equals 0 generate Clear Error in Disconnect Translator (MAC 3.3).
9. Function 1 starts Channel Timing Delay Chain (MAC 3.3).
10. Channel Timing Chain T2 and Clear Error generate Set MAC Clear in Read, Write Control (MAC 3.6).
11. MAC Clear FF sets (MAC 3.6) and outputs MAC Clear Error.
12. MAC Clear Error enters MAC Busy Control translator (MAC 3.7) and activates MAC Read, Write, or Clear if MAC Busy 1 FF and MAC Busy 2 Register are clear (indicating CP is not reading or writing MAC-resident registers).  
  
MAC Clear Error and inactive Instruction Unit Reserve generate Gated Clear Error (MAC 3.6).
13. MAC Read, Write, or Clear generates Start MAC Timing Chain in Start MAC Timing Chain Translator (MAC 3.7).
14. Start MAC Timing Chain enters MAC Timing Chain (MAC 3.7).  
  
MAC Busy 1 FF sets.  
  
MAC Read, Write, or Clear deactivates. Start MAC Timing Chain deactivates.
15. MAC Timing Chain T1 enters PFS, DEC, PTM Board Select Decoders (MAC 3.5) and ANDs with Gated Clear Error.  
  
MAC Busy 2 Register (MAC 3.7) sets.
16. Five inputs to Board Select Register (MAC 3.5) become active and remain active until MAC Timing Chain T4 deactivates (step 24).

17. MAC Timing Chain T2 activates.  
Board Select 0 through 4 (MAC 3.5) activate and proceed to Holding Register in Set/Clear PFS Control (MAC 3.10).
18. Board Select 0 through 4 enter Holding Register and AND with Gated Clear Error to activate Board 0 through 4 Clear Error.
19. Board 0 through 4 Clear Error activates Clear PFS Register 80 through 89 in Set/Clear PFS Control translator.
20. Clear PFS Register 80 through 89 clear PFS Registers 80 through 89 (MAC 3.9).
21. MAC Timing Chain T4 activates.
22. Inactive Instruction Unit Reserve (CP access to MAC-resident registers blocked by Busy 1 FF) and MAC Timing Chain T4 generate Ready In Enable in Disconnect Translator (MAC 3.3).
23. MAC Clear Error and Ready In Enable activate Disconnect In To IOU.
24. MAC Timing Chain T5 activates and, with inactive MAC Response/Request From Pipe, generates Clear MAC Busy 1 in Clear MAC Operation Translator (MAC 3.7).  
  
MAC Timing Chain T5 and inactive Instruction Issue Reserve activate Byte Transfer or Clear Error Complete in Read, Write Control (MAC 3.6).
25. MAC Timing Chain T5 deactivates.  
  
Board Select Register (MAC 3.5) clears.  
  
Clear MAC Busy 1 clears MAC Busy 1 FF (MAC 3.7).
26. MAC Clear Error and Gated Clear Error deactivate.
27. Clear PFS Register 80 through 89 deactivate (MAC 3.10).
28. MAC Busy 2 Register clears.

#### IOU READ, WRITE OPERATIONS

Read, Write, and Echo employ Activate Out and Ready Out control signals, and Control Words 1 and 2 after Function and the Function Code enter MAC. Control Words 1 and 2 supply read or write addresses.

These descriptions are referenced by Read, Write Initial Steps, which follows.

#### ACTIVATE OUT

1. Activate Out arrives at Maintenance Channel Control Signal Resynchronization Network (MAC 3.3).
2. Resynchronized Activate Out departs 48-ns Delay and Resynchronization Network (MAC 3.3).

3. Resynchronized Activate Out enters Resynchronization Register.
4. Activate 1 proceeds to Read and Write FFs translator (MAC 3.4) and Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2), where it partially enables Set Read Operation.
5. Activate 1 clears Control Word 2 FF (MAC 3.3).

#### READY OUT

1. Ready Out arrives at Maintenance Channel Control Signal Resynchronization Network (MAC 3.3).
2. Resynchronized Ready Out departs 48-ns Delay and Resynchronization Network (MAC 3.3).
3. Resynchronized Ready Out enters Resynchronization Register.
4. Ready 1 loads Maintenance Channel Data Register (MAC 3.0).

Ready 1 partially enables Set Write Operation in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2) and Select 1 and 2 in Partial Address Incrementer (MAC 3.2).

Ready 1 partially enables Set Write in Read and Write FFs translator (MAC 3.4), Set MAC Read in Read, Write Control (MAC 3.6), and Start MAC Timing Chain in Start MAC Timing Chain Translator (MAC 3.7).

5. Ready 1 clears Control Word 1 FF (MAC 3.3) and enables set inputs to Control Word 2 and Control Words Loaded FFs.

Ready 1 starts Channel Timing Delay Chain (MAC 3.3).

#### CONTROL WORD 1

1. Control Word 1 arrives at Maintenance Channel Parity Check (MAC 3.0) as Maintenance Channel Data Bits 0 through 7/8.
2. Parity is checked and byte proceeds to Partial Address Incrementer (MAC 3.2).  
(If parity error occurs, Maintenance Channel Input Parity Error 0 enters Maintenance Channel Control Signal Resynchronization Network (MAC 3.3) translator.)
3. Once Ready 1 activates, Select 1 gates Maintenance Channel Data Bits 5 through 7 into Partial Address Incrementer (MAC 3.2).  
(Maintenance Channel Input Parity Error 0, Control Word 1, and Ready 1 generate Write Parity Error (MAC 3.3). Write Parity Error activates PFS Register 86 Bit 33 and Error In To IOU in PFS Translator (MAC 3.8).)

#### CONTROL WORD 2

1. Control Word 2 arrives at Maintenance Channel Parity Check (MAC 3.0) as Maintenance Channel Data Bits 0 through 7/8.

2. Parity is checked and byte proceeds to Partial Address Incrementer (MAC 3.2), Address Incrementer (MAC 3.6) and Address Register (MAC 3.2).

(If parity error occurs, Maintenance Channel Input Parity Error 0 enters Maintenance Channel Control Signal Resynchronization Network (MAC 3.3) translator.)

3. Once Ready 1 activates, Select 2 (Control Word 2 and Ready 1) gates Maintenance Channel Data Bits 0, 1 into Partial Address Incrementer and Bits 0 through 7/8 into Address Register (MAC 3.2).

(Maintenance Channel Input Parity Error 0, Control Word 2, and Ready 1 generate Write Parity Error (MAC 3.3). Write Parity Error activates PFS Register 86 Bit 33 and Error In to IOU in PFS Translator (MAC 3.8).)

Control Word 2 and Ready gate Bits 0 through 7/8 into Address Incrementer (MAC 3.6).

#### READ, WRITE INITIAL STEPS

All read and write data transfers between the IOU and MAC (Opcode equals four, five, or eight) share the following initial steps in their execution.

1. Function and Function Code arrive in MAC. Refer to Function description for details.
2. Function 2 enters Disconnect Translator (MAC 3.3) to generate Disconnect In to IOU.  
Function 1 starts Channel Timing Delay Chain (MAC 3.3) and sets Control Word 1 FF (MAC 3.3).
3. IOU sends Activate Out to MAC. Refer to Activate Out description for details.
4. IOU sends Ready Out and Control Word 1 to MAC. Refer to Ready Out and Control Word 1 descriptions for details.
5. Ready 1 clears Control Word 1 FF, sets Control Word 2 FF (MAC 3.3), and starts Channel Timing Delay Chain (MAC 3.3).
6. Channel Timing Chain T4 and Control Word 2 generate Set Ready In To IOU in Data Ready Control (MAC 3.3) translator.
7. Ready In To IOU proceeds to IOU.
8. IOU sends Ready Out and Control Word 2 to MAC. Refer to Ready Out and Control Word 2 descriptions for details.
9. Ready 1 sets Control Words Loaded FF (MAC 3.3) and starts Channel Timing Delay Chain (MAC 3.3).
10. Channel Timing Chain T4 and Control Word 2 generate Set Ready In To IOU in Data Ready Control (MAC 3.3) translator.
11. Ready In To IOU proceeds to IOU.
12. IOU sends Activate Out to MAC. Refer to Activate Out description for details.

#### ECHO (FUNCTION CODE = 8X)

The Echo operation tests the Maintenance Channel data path between the IOU and MAC. While Echo uses the channel protocol associated with read operations, it does not transmit CP read data to the IOU. Instead, Control Word 2 returns to the IOU following every data request. The IOU determines the duration of the transfer.

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations.

Opcode of eight activates Operation Decode Equals 8 in Opcode Decoder 0 (MAC 3.2). Type Code has no significance in Echo operation.

Operation Decode Equals 8 enters Data Mux 3 Select Control (MAC 3.1) and activates Mux 3 Select 1, 2. Mux 3 Select 1, 2 enable Maintenance Channel Address Bits 0 through 7/8 into Data Mux 3 (MAC 3.1).

Operation Decode Equals 8 enters Read, Write FFs translator (MAC 3.4) to partially enable Set Read.

Only Control Word 2 contains valid data, which enters Address Register (MAC 3.2) as Maintenance Channel Data Bits 0 through 7/8.

Second Activate Out generates Set Read in Read and Write FFs translator (MAC 3.4).

2. Read FF sets (MAC 3.4) and Channel Timing Delay Chain (MAC 3.3) starts. Set Read deactivates.
3. Read or Write partially enables Set Ready In To IOU in Data Ready Control (MAC 3.3) translator.
4. At Channel Timing Chain T4, Set Ready In To IOU activates.
5. Gate Channel Output Register loads Channel Output Register (MAC 3.1) with Maintenance Channel Address Bits 0 through 7/8 from Address Register (MAC 3.2).
6. Set Ready In To IOU enters Data Ready Control Holding Register (MAC 3.3) and departs for IOU as Ready In To IOU.  
  
A parity error detected in Data Mux 3 Parity Check (MAC 3.1) activates PFS Register 86 Bit 32 and Error In to IOU in PFS Translator (MAC 3.8).
7. IOU sends Ready Out to MAC (refer to IOU Read, Write Operations, Ready Out description for details) and starts Channel Timing Delay Chain (MAC 3.3).
8. Steps 4 through 7 repeat until IOU terminates Echo operation.

#### DIRECT READS, WRITES

The IOU can read or write various registers and memories in the CP via MAC without microcode assistance from the CP. Operations of this type are referred to as direct.

MAC reads and writes are direct operations involving MAC-resident registers. Reading or writing MAC-resident registers requires use of conflict-control circuitry in MAC, because the CP also has access to the registers. MAC reads and writes are described separately.

### CMC Maintenance Register Read, Write

The IOU has access to nine maintenance registers in CMC which it selects with the address contained in Control Word 2. Eight of these locations may be read, six written (refer to table 2-1).

#### CMC Maintenance Register Read (Function Code = 4A)

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations.

Opcode of four activates Operation Decode Equals 4 in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Type Code of A activates Type Decode Equals A.

Operation Decode Equals 4 enters Read, Write FFs translator (MAC 3.4) to partially enable Set Read.

Type Decode Equals A activates Mux 2 Select 1, 2 in Data Mux 2 Select Control (MAC 3.1). Mux 2 Select 1, 2 enable CMC Data Bits 0 through 7/8 into Data Mux 2 (MAC 3.1) and Data Mux 2 Bits 0 through 7/8 into Data Mux 3 (MAC 3.1).

Type Decode Equals A partially enables Force Count Equal to 7 in Channel Byte Counter, Decoder (MAC 3.3).

Inactive Gated Type Decode Equals 0 activates Direct in Read and Write FF Translator (MAC 3.4).

Control Word 2, containing CMC maintenance register address, enters Address Register (MAC 3.2) as Maintenance Channel Data Bits 0 through 7/8.

Decoded maintenance register address is sent to CMC as Maintenance Register Select Bits 0 through 3. Maintenance Register Select Bits 0 through 3 select the appropriate read data for output from Upper/Lower Data Mux (CMC 3.14).

Second Activate Out generates Set Read in Read and Write FFs translator (MAC 3.4).

2. Read FF sets (MAC 3.4) and Channel Timing Delay Chain (MAC 3.3) starts.

Set Read deactivates.

3. Read generates Force Count Equal to 7 in Channel Byte Counter, Decoder (MAC 3.3) if CMC maintenance register address in Address Register (MAC 3.2) is  $00_{16}$ . (CMC Status Summary byte is read eight successive times because of forced value in byte counter.)

Byte Selection for all other CMC read operations is controlled by incrementing Channel Byte Count Bits 0 through 2 (MAC 3.3). Counter bits are initially set to zero or Write partially enables Increment in Channel Byte Counter, Decoder (MAC 3.3).

Inactive Write and Channel Byte Count Bits 0 through 2 (equal to 0) gate Upper/Lower Register Bits 0 through 7 (CMC 3.14) to Data Mux 2 (MAC 3.1) as CMC Data Bits 0 through 7/8.

4. At Channel Timing Chain T4, Set Ready In To IOU activates in Data Ready Control (MAC 3.3).

5. Gate Channel Output Register loads CMC data byte into Channel Output Register (MAC 3.1).
6. Ready In To IOU departs Holding Register (MAC 3.3) for IOU, accompanying data byte. Channel Timing Chain T5 increments Channel Byte Counter (MAC 3.3).
7. IOU sends Ready Out to MAC (refer to IOU Read, Write Operations, Ready Out description for details) and starts Channel Timing Delay Chain (MAC 3.3).
8. Steps 4 through 7 repeat until eight bytes have been transmitted from selected CMC maintenance register to IOU.
9. IOU terminates operation.

#### CMC Maintenance Register Write (Function Code = 5A)

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations.  
  
Opcode of five activates Operation Decode Equals 5 in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Type Code of A activates Type Decode Equals A.  
  
Operation Decode Equals 5 enters Read and Write FFs translator (MAC 3.4) to partially enable Set Write.  
  
Inactive Gated Type Decode Equals 0 activates Direct in Read and Write FFs translator.  
  
Control Word 2, containing CMC maintenance register address, enters Address Register (MAC 3.2) as Maintenance Channel Data Bits 0 through 7/8.  
  
Decoded maintenance register address proceeds to CMC as Maintenance Register Select Bits 0 through 3. Maintenance Register Select Bits 0 through 3 enter Write Selector (CMC 3.14) and enable writing of the selected CMC maintenance register byte.
2. IOU sends Ready Out and data byte to MAC. Refer to IOU Read, Write Operations, Ready Out description for details.
3. Ready 1 loads Maintenance Channel Data Bits 0 through 7/8 into Maintenance Channel Data Register (MAC 3.0).  
  
Ready 1 generates Set Write in Read and Write FFs translator.
4. Channel Timing Delay Chain (MAC 3.3) starts.  
  
Write FF (MAC 3.4) sets. Read or Write partially enables Increment in Channel Byte Counter, Decoder (MAC 3.3).
5. At Channel Timing Chain T4, Write (MAC 3.2) proceeds to Byte Selector (CMC 3.14).  
  
Set Ready In To IOU activates in Data Ready Control (MAC 3.3) translator.
6. Combination of Write, Channel Byte Count decode and maintenance register address enables MAC Data Bits 0 through 7/8 to appropriate CMC maintenance register (Write Selector, CMC 3.14).

7. Channel Timing Chain T5 increments Channel Byte Counter (MAC 3.3).

Ready In To IOU departs Data Ready Control for IOU.

8. Steps 2 through 7 repeat until eight bytes have been sent to CMC maintenance register. (Variation, step 4 - only portion of step to repeat is starting of Channel Timing Delay Chain. Write FF remains set for duration of write operation.)
9. IOU terminates operation.

CP Status Summary Read (Function Code = 40, Address = 00)

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations.

Opcode of four activates Operation Decode Equals 4 in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Gated Type Decode Equals 0 also activates.

Operation Decode Equals 4 enters Read and Write FFs translator (MAC 3.4) to partially enable Set Read.

Gated Type Decode Equals 0 partially enables Mux 3 Select 2 in Data Mux 3 Select Control (MAC 3.1).

Control Word 2, containing register address  $00_{16}$ , enters Address Register (MAC 3.2) as Maintenance Channel Data Bits 0 through 7/8. Maintenance Channel Address Equals 0X activates Mux 3 Select 2 and Direct (Read and Write FFs, MAC 3.4).

Second Activate Out generates Set Read in Read and Write FFs translator.

2. Read FF sets (MAC 3.4) and Set Read deactivates.

Channel Timing Delay Chain starts.

3. At Channel Timing Chain T4, Set Ready In To IOU activates in Data Ready Control (MAC 3.3) translator.
4. Gate Channel Output Register loads CP status summary byte into Channel Output Register (MAC 3.1). Status summary byte enters register by way of Data Muxes 1 through 3 (MAC 3.1).
5. Ready In To IOU departs Holding Register (MAC 3.3) for IOU, accompanying data byte.
6. IOU issues Ready Out and starts Channel Timing Delay Chain.
7. Steps 3 through 6 repeat until status summary byte has been transferred to IOU eight successive times.
8. IOU terminates operation.



CST Micrand Address Register, Breakpoint Register Read (Function Code = 40, Address = 31, 32)

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations.

Opcode of four activates Operation Decode Equals 4 in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Type Code of zero activates Gated Type Decode Equals 0.

Operation Decode Equals 4 enters Read and Write FFs translator (MAC 3.4) to partially enable Set Read.

Gated Type Decode Equals 0 enters Data Mux 2 Select Control (MAC 3.1) and partially enables CST Address or Breakpoint Register.

Control Word 2, containing register address 31 (MAR) or 32 (Breakpoint), enters Address Register (MAC 3.2) as Maintenance Channel Data Bits 0 through 7/8.

Maintenance Channel Address Equals 3X and Maintenance Channel Address Equals X1 or X2 enter Data Mux 2 Select Control to activate CST Address or Breakpoint Register.

CST Address or Breakpoint Register activates Mux 2 Select 2 to select CST Micrand Address Register/Breakpoint Bits 0 through 7/8 in Data Mux 2 (MAC 3.1).

Mux 2 Select 2 activates Mux 3 Select 2 in Data Mux 3 Select Control (MAC 3.1). Mux 3 Select 2 selects Data Mux 2 Bits 0 through 7/8 in Data Mux 3 (MAC 3.1).

CST Address or Breakpoint Register also activates Byte 0 through 5 Zero Fill in CST Register Zero Fill Translator (MAC 3.1), forcing zeros to Channel Output Register input (MAC 3.1). Bytes 0 through 5 of data transfer are zero-filled. MAR and Breakpoint Register data are transferred in byte six and seven positions.

CST Address or Breakpoint Register partially enables CST Opcode Bits 0, 1 in CST Interface (MAC 3.4) and activates Direct in Read and Write FFs translator (MAC 3.4).

Second Activate Out generates Set Read in Read and Write FFs translator.

2. Read FF sets (MAC 3.4) and Set Read deactivates.

Channel Timing Delay Chain (MAC 3.3) starts.

3. Read Or Write partially enables Increment in Channel Byte Counter, Decoder (MAC 3.3).

CST Opcode Bits 0, 1 activate in CST Interface (decode of two denotes Breakpoint Register, decode of three denotes Micrand Address Register).

4. If decode equals two, Select Breakpoint Register activates in MAC Off Line Control (CST 3.0)

5. At Channel Timing Chain T4, Set Ready In To IOU activates in Data Ready Control (MAC 3.3).

6. Gate Channel Output Register loads zeros into Channel Output Register (MAC 3.1).

7. Ready In To IOU departs Holding Register (MAC 3.3) for IOU, accompanying data byte.

Channel Timing Chain T5 increments Channel Byte Counter (MAC 3.3).

8. IOU sends Ready Out to MAC (refer to IOU Read Write Operations, Ready Out description for details) and starts Channel Timing Delay Chain (MAC 3.3).
9. Steps 5 through 8 repeat until Channel Byte Count equals six.
10. At Channel Timing Chain T5 of byte five transfer, Channel Byte Counter increments to six. Decode of six disables Byte 0 through 5 Zero Fill in CST Register Zero Fill Translator (MAC 3.1).  
  
Micrand Address Register Bits 0 through 2 or Breakpoint Register Bits 0 through 2 are selected in Micrand Address Register/Breakpoint Register Disassembly Mux (CST 3.3). Data Mux 3 (MAC 3.1) selects CST Micrand Address Register/Breakpoint Bits 0 through 7/8.
11. IOU sends Ready Out to MAC (byte six request) and starts Channel Timing Delay Chain. Steps 5 through 8 repeat, with first Micrand Address Register/Breakpoint byte sent to IOU in place of zeros.
12. After Channel Byte Count increments to seven, Rightmost Byte activates in Channel Byte Counter, Decoder (MAC 3.3).  
  
Micrand Address Register Bits 3 through 10 or Breakpoint Register Bits 3 through 10 are selected in Micrand Address Register/Breakpoint Register Disassembly Mux as final byte of transfer.
13. IOU terminates operation.

CST Micrand Address Register, Breakpoint Register Write (Function Code = 50, Address = 31, 32)

MAC must be operating off line. No Start operation may have occurred to place the CP on line.

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations.

Opcode of five activates Operation Decode Equals 5 in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Type Code of zero activates Gates Type Decode Equals 0.

Operation Decode Equals 5 enters Read and Write FFs translator (MAC 3.4) to partially enable Set Write.

Gated Type Decode Equals 0 enters Data Mux 2 Select Control (MAC 3.1) and partially enables CST Address or Breakpoint Register.

Control Word 2, containing register address 31 or 32, enters Address Register (MAC 3.2) as Maintenance Channel Data Bits 0 through 7/8.

Maintenance Channel Address Equals 3X and Maintenance Channel Address Equals X1 or X2 enter Data Mux 2 Select Control to activate CST Address or Breakpoint Register.

CST Address or Breakpoint Register activates Direct in Read and Write FFs translator (MAC 3.4). CST Address or Breakpoint Register also partially enables CST Opcode Bits 0, 1 and Clock Microcode in CST Interface (MAC 3.4).

2. IOU sends Ready Out and data byte to MAC. Refer to IOU Read, Write Operations, Ready Out description for details.

Maintenance Channel Data Bits 0 through 7/8 from IOU enter Microcode Request Control Mux (MAC 3.6) and depart for MAC Left Most Byte Mux Register (CST 3.3) as MAC Microcode Address Bits 0 through 7/8.

3. Ready 1 generates Set Write in Read and Write FFs translator (MAC 3.4).
4. Channel Timing Delay Chain starts.

Write FF (MAC 3.4) sets.

5. Write partially enables Clock Microcode in CST Interface (MAC 3.4).

Read or Write also partially enables Clock Microcode and activates CST Opcode Bits 0, 1. Decode of two denotes Breakpoint Register, decode of three denotes Micrand Address Register.

Read Or Write partially enables Increment in Channel Byte Counter Decoder (MAC 3.3).

6. If CST Opcode Bits 0, 1 decode equals two, Select Breakpoint Register activates in MAC Off Line Control (CST 3.0) translator. Select Breakpoint Register partially enables Load Breakpoint Register translator.

Decode of three partially enables Load Micrand Address Mux Register in MAC Off Line Control translator.

7. At Channel Timing Chain T4, Set Ready In To IOU activates in Data Ready Control (MAC 3.3).

8. Channel Timing Chain T5 increments Channel Byte Counter (MAC 3.3).

Ready In To IOU reaches the Maintenance Channel from Data Ready Control.

9. IOU sends Ready Out and data byte and starts Channel Timing Delay Chain.

10. Steps 7 through 9 repeat until Channel Byte Count equals six. First six bytes of write operation are invalid and not retained by hardware.

11. At Channel Timing Chain T5 of sixth byte transfer, Channel Byte Counter increments to six.

12. Channel Byte Count Decode 6 enters CST Register Zero Fill Translator (MAC 3.1) and generates Channel Byte Count Decode Equals 6 or 7.

13. Channel Byte Count Decode Equals 6 or 7 partially enables Clock Microcode in CST Interface (MAC 3.4).

14. IOU sends Ready Out and next data byte to MAC.

15. Channel Timing Delay Chain starts.

16. At Channel Timing Chain T4, Set Ready In To IOU activates in Data Ready Control (MAC 3.3).

17. Channel Timing Chain T5 activates Clock Microcode in CST Interface.

Channel Timing Chain T5 increments Channel Byte Counter (MAC 3.3).

Ready In To IOU departs Data Ready Control.

18. Clock Microcode activates Load Leftmost Byte in MAC Off Line Control (CST 3.0).

Load Leftmost Byte loads MAC Microcode Address Bits 5 through 7/8 into MAC Leftmost Byte Mux Register (CST 3.3). Bits 5 through 7 enter positions 0 through 2 of the register.

19. IOU sends Ready Out and final data byte to MAC.

20. Channel Timing Delay Chain starts.

21. Channel Timing Chain T5 activates Clock Microcode in CST Interface.

22. Right Most Byte (Channel Byte Count Bit 2) and Clock Microcode activate Load Rightmost Byte.

23. Load Breakpoint Register activates if CST Opcode Bits 0, 1 equal two. Load Micrand Address Mux Register activates if Opcode bits equal three.

24. Load Breakpoint Register loads MAC Microcode Address Bits 0 through 10/Parity into Breakpoint Register (CST 3.3), or Enable Micrand Address Mux Register loads MAC Microcode Address Bits 0 through 10/Parity into Micrand Address Mux Register (CST 3.4). Data path to register is by way of Micrand Address Muxes 2, 1 (CST 3.4).

25. Micrand Address Mux Register Bits 0 through 10/Parity enter Micrand Address Register.

26. IOU terminates operation.

#### CST RAM Read, Write

The CST RAM of models AD112-C and AD113-A is divided into two memories of 2048K locations each. The second memory is a duplicate of the first and is executed when a parity error occurs. Models AD112-A, B do not have the shadow memory feature. These models have a CST RAM with 2,048 memory locations. In all models each memory location contains a 128-bit micrand used to control execution of C170 and C180 product-set instructions. When the IOU performs a CST RAM read or write, the steps are similar to those of direct reads and writes described earlier. A major difference is that Control Words 1 and 2 do not supply the CST RAM address. The address first must be placed in the CST Micrand Address Mux Register by a Function Code of 50 and an address of 31 (CST 3.4). CST RAM data transfers are not limited to eight bytes.

Because micrands contain 128 bits, the CST RAM address increments once for every 16 bytes transferred.

MAC must be operating off line. No Start operation may have occurred to place the CP on line.

#### CST RAM Read (Function Code = 41)

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations.

Opcode of four activates Operation Decode Equals 4 in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Type Code of one activates Type Decode Equals 1 and Gate Type Decode Equals 1.

Operation Decode Equals 4 enters Read and Write FFs translator (MAC 3.4) to partially enable Set Read.

Type Decode Equals 1 and MAC Off Line (MAC is off line between CP Master Clear and Start operations) disable Select Register File to Disassembly Network in CST Disassembly Control (MAC 3.8).

Type Decode Equals 1 and MAC Off Line partially enable Register File Go in Register File Go Control (MAC 3.8).

Type Decode Equals 1 or 7 deactivates Select PFS Bus in CST Disassembly Control (MAC 3.8), which in turn enables CST Byte Count Bit 0 to CST Disassembly Network (CST 3.5).

Gated Type Decode Equals 1 partially enables Toggle, CST Opcode Bit 1 and Clock Microcode in the CST Interface (MAC 3.4). Direct activates in Read and Write FFs translator (MAC 3.4).

Second Activate Out generates Set Read Operation in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2).

Set Read activates in Read and Write FFs translator.

2. Read FF sets (MAC 3.4), and Set Read deactivates.

Channel Timing Delay Chain starts (MAC 3.3).

Set Read Operation sets Control FF (MAC 3.2).

3. Read Operation generates Register File Go in Register File Go Control (MAC 3.8).

CST Opcode Bit 1 activates in CST Interface (MAC 3.4) translator.

Read or Write partially enables Increment in Channel Byte Counter, Decoder (MAC 3.3) and Increment Counter in MAC Byte Counter (MAC 3.8).

4. MAC Opcode Equals Increment and MAC Load or Dump activate in MAC Off Line Control (CST 3.0). CST Opcode of one partially enables Load Micrand Address Mux Register.
5. MAC Load or Dump disables static set input to Functional Unit Micrand FF (CST 3.0).

MAC Opcode Equals Increment and inactive On Line generate Plus 1 in Micrand Address Mux 1 Select (CST 3.4).

6. Select 1 activates, creating path for incremented Micrand Address Mux Register Bits to enter Micrand Address Mux Register (CST 3.4) later. Refer to step 25.

7. Register File Go leaves Holding Register (MAC 3.8) and enters CST Disassembly Network Interface (OPI 3.4). Register File Go generates Strobe Disassembly Register, a continuous input to Disassembly Register (CST 3.5).

8. General Micrand enters Disassembly Register.

General Micrand, high-order 64 bits of 128-bit Micrand addressed by contents of Micrand Address Mux Register, are present at Disassembly Register (CST 3.5) input after Select Register File To Disassembly Network deactivates (step 1). Statically active Advance Micrand, Functional Unit Advance and inactive Select Functional Unit Micrand enable General Micrand's path from CST RAM (CST 3.5) to Disassembly Network.

9. CST Byte Count Bits 0 through 2 equal to zero select high-order byte in Disassembly Muxes 1 and 2 for output to Data Mux 5 (MAC 3.0) as MAC Bus Data Bits 0 through 7/8.
10. Data Mux 5 Bits 0 through 7/8 enter Data Mux 3 (MAC 3.1) and proceed to Channel Output Register. (Even though Control Words 1 and 2 are not used, care must be taken in selecting values to prevent Mux 5 Select 1 from deactivating or Mux 5 Select 2 from activating due to false address decodes.)
11. At Channel Timing Chain T4, Set Ready In To IOU activates in Data Ready Control translator (MAC 3.3).
12. Gate Channel Output Register loads Channel Output Register (MAC 3.1).
13. Ready In To IOU proceeds from Holding Register (MAC 3.3) to IOU, accompanying data byte.  
  
Channel Timing Chain T5 increments Channel Byte Counter (MAC 3.3) and MAC Byte Counter (MAC 3.8). MAC Byte Counter Bits 0 through 2 depart CST Disassembly Control (MAC 3.8) as CST Byte Count.
14. IOU sends Ready Out to MAC (refer to IOU Read, Write Operations, Ready out description for details) and starts Channel Timing Delay Chain.
15. Steps 11 through 14 repeat. Incremented CST Byte Count selects new byte each cycle in CST Disassembly Network (CST 3.5) until eight bytes depart MAC for IOU.
16. After Channel Byte Count reaches seven (last byte of General Micrand), Channel Timing Chain T5 activates Toggle in CST Interface translator (MAC 3.4) and presets byte counters to zero.
17. Toggle Register (MAC 3.4) sets.
18. Select Functional Unit Micrand departs CST Interface (MAC 3.4) to set Functional Unit Micrand FF (CST 3.0). Functional Unit Micrand FF remains set for next eight-byte transfer.
19. Functional Unit Micrand Bits 0 through 63/64 through 71 enter RAM Data Mux (CST 3.5) and proceed to Disassembly Register (CST 3.5) by way of Functional Unit Micrand Buffer Register and Functional Unit Micrand Register.
20. IOU sends Ready Out to MAC and starts Channel Timing Delay Chain.
21. Steps 11 through 14 repeat for next eight micrand bytes.
22. After byte count reaches seven (last byte of Functional Unit Micrand), Channel Timing Chain T5 activates Toggle and Clock Microcode in CST Interface Translator and presets byte counters to zero.
23. Clock Microcode activates Load Micrand Address Mux Register in MAC Off Line Control translator (CST 3.0). Enable Micrand Address Mux Register activates.
24. Toggle Register sets.
25. Next CST RAM address (Incremented Micrand Address Register Bits 0 through 10/Parity) passes through Micrand Address Mux 1 (CST 3.4) and enters Micrand Address Mux Register (CST 3.4).
26. CST Address Bits 0 through 10 address new micrand in CST RAM.

27. Select Functional Unit Micrand deactivates in CST Interface (MAC 3.4).
28. Functional Unit Micrand FF clears (CST 3.0).
29. General Micrand Bits 0 through 63/64 through 71 from new CST RAM address enter RAM Data Mux (CST 3.5). Disassembly and transfer of new 128-bit micrand begins.
30. Addressing and disassembly of micrands continue until IOU terminates operation.

#### CST RAM Write (Function Code = 51)

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations.

Opcode of five activates Operation Decode Equals 5 in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Type Code of one activates Type Decode Equals 1 and Gated Type Decode Equals 1.

Operation Decode Equals 5 enters Read and Write FFs translator (MAC 3.4) to partially enable Set Write. Inactive Gated Type Decode Equals 0 activates Direct.

Gated Type Decode Equals 1 partially enables Toggle, CST Opcode Bit 1, Clock Microcode and Interface Enable in CST Interface (MAC 3.4).

2. IOU sends Ready Out and data byte to MAC. Refer to IOU Read, Write Operations, Ready Out description for details.

Data byte enters Maintenance Channel Data Parity Check (MAC 3.0) as Maintenance Channel Data Bits 0 through 7/8.

3. Maintenance Channel Data Bits 0 through 7/8 proceed to CST RAM (CST 3.5), where parity is checked again.
4. Ready 1 generates Set Write in Read and Write FFs translator (MAC 3.4).
5. Channel Timing Delay Chain (MAC 3.3) starts.

Write FF (MAC 3.4) sets.

6. Read or Write partially enables Increment in Channel Byte Counter, Decoder (MAC 3.3).

Write partially enables Channel Byte Count Equals 2, 3, 6, or 7 in CST Interface translator (MAC 3.4).

CST Opcode Bit 1 activates in CST Interface.

7. MAC Opcode Equals Increment and MAC Load or Dump activate in MAC Off Line Control (CST 3.0).

CST Opcode Bit 1 partially enables Load Micrand Address Mux Register.

8. MAC Load or Dump disables static set input to Functional Unit Micrand FF (CST 3.0).

MAC Opcode Equals Increment and inactive On Line generate Plus 1 in Micrand Address Mux 1 Select (CST 3.4).

9. Select 1 activates, creating path for incremented Micrand Address MUX Register Bits to enter Micrand Address Mux Register (CST 3.4) later. Refer to step 26.
10. At Channel Timing Chain T4, Set Ready In To IOU activates in Data Ready Control translator (MAC 3.3).

Interface Enable activates in CST Interface translator (MAC 3.4).

11. After Decoded Channel Byte Count Bits 0 through 2 are decoded, CST Write Enable 0 through 3 proceed to CST Write Control (CST 3.5).
12. Channel Byte Count Equals 2, 3, 6, or 7, CST Write Enable 0 through 3, and inactive Select Functional Unit Micrand generate Write General Micrand Byte N in CST Write Control (CST 3.5).
13. Maintenance Channel Data Bits 0 through 7/8 enter CST RAM location addressed by contents of Micrand Address Mux Register.
14. Channel Timing Chain T5 increments Channel Byte Counter (MAC 3.3).  
Ready In To IOU departs Data Ready Control (MAC 3.3).
15. IOU sends Ready Out and data byte to MAC and starts Channel Timing Delay Chain.
16. Steps 10 to 15 repeat until eight bytes enter addressed CST RAM location. Incremented Channel Byte Count loads new byte into addressed RAM location each cycle.
17. After byte count reaches seven (indicating transfer of General Micrand's last byte), Channel Timing Chain T5 activates Toggle in CST Interface translator (MAC 3.4) and presets byte counter to zero.
18. Toggle Register sets.
19. Select Functional Unit Micrand from CST Interface (MAC 3.4) sets Functional Unit Micrand FF (CST 3.0). Functional Unit Micrand FF remains set for next eight-byte transfer.
20. IOU sends Ready Out and data byte to MAC and starts Channel Timing Delay Chain.
21. Steps 10 to 15 repeat for next eight bytes. (Variation, step 12 - Channel Byte Count Equals 2, 3, 6, or 7, CST Write Enable 0 through 3, and Select Functional Unit Micrand generate Write Functional Unit Micrand Byte N instead of Write General Micrand Byte N in CST Write Control (CST 3.5).)  
Incremented Channel Byte Count loads new byte into addressed RAM location each cycle.
22. After byte count reaches seven for second time (last byte of Functional Unit Micrand), Channel Timing Chain T5 activates Toggle and Clock Microcode in CST Interface translator. It also presets byte counter to zero.
23. Clock Microcode activates Load Micrand Address Mux Register in MAC Off Line Control translator (CST 3.0).
24. Enable Micrand Address Mux Register activates.
25. Toggle Register sets.



26. Next CST RAM address (Incremented Micrand Address Register Bits 0 through 10/Parity) passes through Micrand Address Mux 1 (CST 3.4) and enters Micrand Address Mux Register (CST 3.4).
27. CST Address Bits 0 through 10 address new location in CST RAM.
28. Select Functional Unit Micrand deactivates in CST Interface (MAC 3.4).
29. Functional Unit Micrand FF clears (CST 3.0).

Writing into next 128-bit micrand address begins, with Write General Micrand Byte 0 through 7 controlling input of first eight bytes into CST RAM.

30. Addressing and writing of CST RAM continues until IOU terminates operation.

#### Control Memory Read, Write

Several RAMs in the CP are the functional equivalent of ROMs. They are used in place of ROMs because they may be accessed more quickly. It is necessary to load these Control Memory RAMs during CP initialization.

Included are the Reference ROM (Type Code equals three), Soft Control Memories (Type Code equals four), BDP Memories (Type Code equals five), and First Level Instruction Decoder (Type Code equals six). Although these memories vary in width, all are considered by the IOU to contain four bytes per address. Unused bytes are zero-filled.

Control Words 1 and 2 provide the read and write addresses.

The address increments for every four bytes (half-word) transferred. The amount of data contained in each RAM varies, with unused locations zero-filled.

MAC must be operating off line in all cases except for reading or writing the Reference ROM. No Start operation may have occurred to place the CP on line.

Reference ROM Write (Function Code = 53)

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations. Refer to figure 2-2 for contents of Control Words 1 and 2.

Opcode of five activates Operation Decode Equals 5 in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Type Code of three activates Type Decode Equals 3 and Memory Type 3, 4, 5, or 6.

Operation Decode Equals 5 enters Read and Write FFs translator (MAC 3.4) to partially enable Set Write. Inactive Gated Type Decode Equals 0 activates Direct.

Type Decode Equals 3 partially enables control of Reference ROM (MAC 3.6) write data input. Memory Type 3, 4, 5, or 6 partially enables Advance in Address Increment Translator (MAC 3.6).

Control Word 2, containing Reference ROM write address, enters Address Incrementer (MAC 3.6) and proceeds to Reference ROM because MAC is off line.

2. IOU sends Ready Out and data byte to MAC. Refer to Read, Write Operations, Ready Out description for details.

Data byte enters Maintenance Channel Data Parity Check (MAC 3.0) as Maintenance Channel Data Bits 0 through 7/8.

3. Maintenance Channel Data Bits 2 through 7/8 proceed to Reference ROM data input (MAC 3.6).
4. Ready 1 generates Set Write in Read and Write FFs translator (MAC 3.4) and Set Write Operation in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2).
5. Channel Timing Delay Chain (MAC 3.3) starts.

Write FF sets (MAC 3.4).

6. Write Operation departs Control FF (MAC 3.2) to enable decode of MAC Byte Counter bits (MAC 3.8).

Read or Write partially enables Increment Counter in MAC Byte Counter.

7. Channel Timing Chain T3 partially enables control of Reference ROM (MAC 3.6) write data input. Because MAC Byte Count does not equal three or seven, data byte does not enter Reference ROM.

8. Channel Timing Chain T4 activates Set Ready In To IOU in Data Ready Control (MAC 3.3).

9. Channel Timing Chain T5 increments MAC Byte Counter (MAC 3.8).

Ready In To IOU departs Data Ready Control for IOU.

10. IOU sends Ready Out and data byte to MAC, and starts Channel Timing Delay Chain.

11. Steps 7 to 10 repeat until MAC Byte Counter increments to three.

12. IOU sends Ready Out and fourth data byte to MAC and starts Channel Timing Delay Chain.

13. Channel Timing Chain T3 and MAC Byte Count Equals 3 or 7 gate Maintenance Channel Data Bits 2 through 7/8 into Reference ROM (MAC 3.6) location addressed by Reference ROM Address Bits 0 through 7.

14. Channel Timing Chain T4 activates Set Ready In To IOU in Data Ready Control (MAC 3.3).

15. Advance activates in Address Increment Translator (MAC 3.6) and increments MAC Address Bits 0 through 7/8 in Address Incrementer (MAC 3.6).

Channel Timing Chain T5 increments MAC Byte Counter (MAC 3.8).

Ready In To IOU departs Data Ready Control for IOU.

16. IOU sends Ready Out and next data byte to MAC and starts Channel Timing Delay Chain.

17. Steps 7 through 10 repeat until MAC Byte Counter increments to seven, after which steps 12 through 15 repeat to load data into next Reference ROM address.

18. Step 16 repeats, followed by steps 7 through 17 to load data into the next two Reference ROM addresses. Procedure continues until all 256 locations have been loaded.

Figure 2-2 shows address fields and data format for Reference ROM writes.

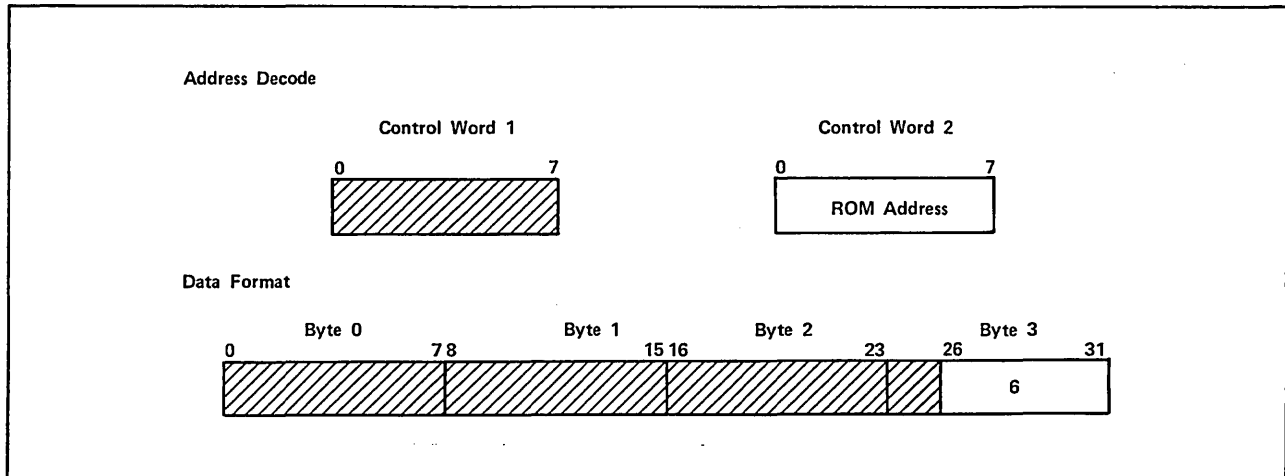


Figure 2-2. Type Code 3 - Reference ROM Address, Data Formats

#### Soft Control Memory Write (Function Code = 54)

A single operation writes three separate soft control RAMs, AC Soft Control Memories 1 and 2, and the ALN Soft Control Memory RAM. The two high-order bits of the write address in the Address Incrementer (MAC 3.6) select the RAMs in sequence.

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations. Refer to figure 2-3 for contents of Control Words 1 and 2.

Opcode of five activates Operation Decode Equals 5 in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Type Code of four activates Type Decode Equals 4, Gated Type Decode Equals 4, and Memory Type 3, 4, 5, or 6.

Operation Decode Equals 5 enters Read and Write FFs translator (MAC 3.4) to partially enable Set Write. Inactive Gated Type Decode Equals 0 activates Direct.

Type Decode Equals 4 enters MAC Byte Selector (MAC 3.8) to partially enable MAC Byte 1 and 3 Select. Memory Type 3, 4, 5, or 6 partially enables Advance in Address Increment Translator (MAC 3.6). Gated Type Decode Equals 4 partially enables Address Increment Bits 8, 9 to Soft Control Go Decoder (MAC 3.2).

Control Word 2, containing RAM address in bit positions 2 through 7 and Soft Control select code in bit positions 0, 1 enters Address Incrementer (MAC 3.6) as Maintenance Channel Data Bits 0 through 7/8. Bits 0, 1 also enter Partial Address Incrementer as Bits 8, 9 (MAC 3.2). MAC Address Bits 0 through 7/8 depart Address Incrementer on MAC Address Bus (MAC 3.6) for Soft Control Memories.

2. IOU sends Ready Out and data byte to MAC. Refer to IOU Read, Write Operations, Ready Out description.

Data byte enters Maintenance Channel Data Parity Check (MAC 3.0) as Maintenance Channel Data Bits 0 through 7/8 and proceeds to Data Mux 6 (MAC 3.0). Bit 8 of even byte enters MAC Even Byte Parity Register.

3. MAC Data Bus Bits 0 through 7/8, 9 depart Data Mux 6 for Soft Control 2 RAM (AC 3.2).
4. Ready 1 generates Set Write in Read and Write FFs translator (MAC 3.4) and Set Write Operation in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2).
5. Channel Timing Delay Chain starts (MAC 3.3).

Write FF sets (MAC 3.4).

6. Write Operation from Control FF (MAC 3.2) enables decode of MAC Byte Counter bits (MAC 3.8).

Read or Write partially enables Address Increment Bits 8, 9 to Soft Control Go Decoder (MAC 3.2) and partially enables Increment Counter in MAC Byte Counter translator (MAC 3.8).

MAC Byte Count of zero activates MAC Byte 0, 1 Select in MAC Byte Selector (MAC 3.8).

7. At Channel Timing Chain T4, Set Ready In To IOU activates in Data Ready Control translator (MAC 3.3).
8. Channel Timing Chain T4 gates Address Increment Bits 8, 9 to Soft Control Go Decoder, where Soft Control Go Select 1 activates.
9. Soft Control Go Select 1 gates MAC Bytes 0, 1 Select to Soft Control 2 Load Control Register (AC 3.2).
10. Soft Control Go Select 1 enters Soft Control 2 Go Control Holding Register (AC 3.2).
11. MAC Address Bits 2 through 7 enter Soft Control 2 Address Register (AC 3.2) because MAC is off line.

MAC Bytes 0, 1 Select enter Soft Control 2 Load Control Register (AC 3.2).

12. Write Soft Control 2 Bytes 0, 1 gate MAC Data Bus Bits 0 through 7/8, 9 into Byte 0 and Byte 1 of designated address in Soft Control 2 RAM.
13. Channel Timing Chain T5 increments MAC Byte Counter (MAC 3.8).

Ready In To IOU departs Data Ready Control for IOU.

14. MAC Byte 0 Select deactivates in MAC Byte Selector (MAC 3.8).
15. IOU sends Ready Out and next data byte to MAC and starts Channel Timing Delay Chain.
16. Steps 7 through 12 repeat. (Variation, step 11 - only MAC Byte 1 Select enters Soft Control 2 Load Control Register (AC 3.2).)

Write Soft Control Byte 1 gates MAC Data Bus Bits 0 through 7/8, 9 into byte 1 of designated address in Soft Control 2 RAM. Bit 8 is previous byte's parity, latched in MAC Even Byte Parity Register (MAC 3.0).

Odd and even byte parity are stored in RAM location with odd data byte. Parity for even byte enters odd byte RAM location during even byte write. Locking parity for odd byte parity out of bit 8 position in MAC Even Byte Parity Register preserves even byte parity already in RAM when odd byte and parity (bit 9) enter.

17. Channel Timing Chain T5 increments MAC Byte Counter (MAC 3.8).

Data Ready Control sends Ready In To IOU to IOU.

18. MAC Byte 2 and 3 Select activate in MAC Byte Selector (MAC 3.8).

19. Step 15 repeats.

20. Steps 7 through 12 repeat. (Variation, step 11 - MAC Byte 2 and 3 Select enter Soft Control 2 Load Control Register (AC 3.2).)

Write Soft Control Bytes 2, 3 gate MAC Data Bus Bits 0 through 7/8, 9 into Byte 2 and 3 of designated address in Soft Control 2 RAM.

21. Channel Timing Chain T5 increments MAC Byte Counter (MAC 3.8) to three.

Ready In To IOU departs Data Ready Control for IOU.

22. MAC Byte 2 Select deactivates in MAC Byte Selector (MAC 3.8).

23. Step 15 repeats.

24. Steps 7 through 12 repeat (Variation, step 11 - only MAC Byte 3 Select enters Soft Control 2 Load Control Register (AC 3.2).)

Write Soft Control Byte 3 gates MAC Data Bus Bits 0 through 7/8, 9 into byte 3 of designated address in Soft Control 2 RAM. Bit 8 is even byte parity, latched in MAC Even Byte Parity Register (MAC 3.0).

25. Channel Timing Chain T5 activates Advance in Address Increment Translator (MAC 3.6).

Data Ready Control sends Ready In To IOU to IOU.

26. MAC Address Bits 0 through 7/8 increment in Address Incrementer (MAC 3.6).

MAC Byte Counter (MAC 3.8) increments to four.

27. MAC Byte 0 and 1 Select activates in MAC Byte Selector (MAC 3.8).

Next four bytes enter new address in Soft Control 2 RAM in manner described above. Afterwards, next RAM location is addressed by incremented MAC Address Bits. MAC Byte Counter returns to zero. Loading of Soft Control 2 RAM continues until MAC Address Bit 2 switches from 1 to 0 (count equals 65) and increments Partial Address Incrementer (MAC 3.2).

During next byte transfer, Soft Control Go Select 2 activates in Soft Control Go Decoder (MAC 3.2). Refer to step 8. Loading of Soft Control 1 RAM (AC 3.1) begins and follows pattern of writing into Soft Control 2 RAM.

When MAC Address Bit 2 switches from 1 to 0 again, Partial Address Incrementer increments. During next byte transfer, Soft Control Go Select 3 activates in Soft Control Go Decoder. Refer to step 8. Loading of Soft Control Memory RAM (ALN 3.1) begins and follows pattern of writing into Soft Control 2 RAM.

28. IOU terminates operation.

Figure 2-3 shows address fields and data format for Soft Control Memory writes.

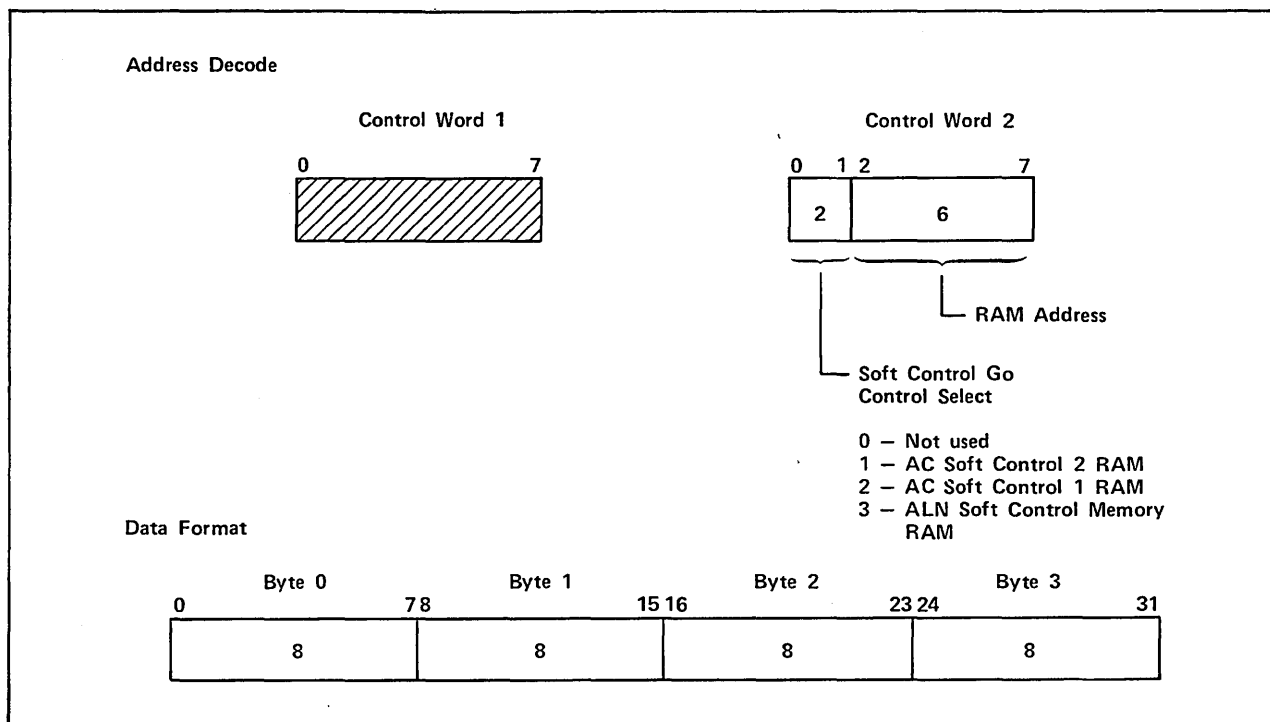


Figure 2-3. Type Code 4 - Soft Control Memories Address, Data Formats

#### BDP Memory Write (Function Code = 55)

A BDP Memory Write operation places data in 2048 32-bit memory locations, divided equally among six 256-word RAMs. RAM addresses provided by the Address Incrementer (MAC 3.6) specify storage locations within individual RAMs. The three low-order bits in the Partial Address Incrementer (MAC 3.2) determine the sequence in which the six RAMs are written.

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations. Refer to figure 2-4 for contents of Control Words 1 and 2.

Opcode of five activates Operation Decode Equals 5 in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Type Code of five activates Gated Type Decode Equals 5 and Memory Type 3, 4, 5, or 6.

Operation Decode Equals 5 enters Read and Write FFs translator (MAC 3.4) to partially enable Set Write. Inactive Gated Type Decode Equals 0 activates Direct.

Gated Type Decode Equals 5 partially enables Address Increment Bits 5 through 7 into BDP Control Memory Go Decoder (MAC 3.2).

Memory Type 3, 4, 5, or 6 partially enables Advance in Address Increment Translator.

Control Word 1, containing BDP Go Select code in bits 5 through 7, enters Partial Address Incrementer (MAC 3.2) as Maintenance Channel Data.

Control Word 2, containing RAM address, enters Address Incrementer (MAC 3.6) as Maintenance Channel Data Bits 0 through 7/8. Bits 0, 1 of RAM address also enter Partial Address Incrementer as Bits 8, 9. MAC Address Bits 0 through 7/8 depart Address Incrementer for BDP Memories on MAC Address Bus (MAC 3.6).

2. IOU sends Ready Out and data byte to MAC. Refer to IOU Read, Write Operations, Ready Out description.

Data byte enters Maintenance Channel Data Parity Check (MAC 3.0) as Maintenance Channel Data Bits 0 through 7/8 and proceeds to Data Mux 6 (MAC 3.0).

3. MAC Data Bus Bits 0 through 7/9 depart Data Mux 6 for A Stream EBCDIC Convert RAM (BDP 3.5).
4. Ready 1 generates Set Write in Read and Write FFs translator (MAC 3.4) and Set Write Operation in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2).
5. Channel Timing Delay Chain starts (MAC 3.3).

Write FF sets (MAC 3.4).

6. Write Operation from Control FF (MAC 3.2) enables decode of MAC Byte Counter bits (MAC 3.8).

Read or Write partially enables Address Increment Bits 5 through 7 to BDP Control Memory Go Decoder (MAC 3.2) and partially enables Increment Counter in MAC Byte Counter translator (MAC 3.8).

MAC Byte Count of zero activates MAC Byte 0 Select in MAC Byte Selector (MAC 3.8).

7. At Channel Timing Chain T4, Set Ready In To IOU activates in Data Ready Control translator (MAC 3.3).

Address Increment Bits 5 through 7 enter BDP Control Memory Go Decoder (MAC 3.2), where BDP Go Select 0 activates.

8. BDP Go Select 0 enters 4 Clock Delay (BDP 3.5) and gates MAC Address Bits 0 through 7/8 through A Stream Data Mux (BDP 3.5).
9. A Stream Data Bits 0 through 7/8 enter A Stream Stage 1 Register (BDP 3.5).
10. Data Bits 0 through 7/32 address A Stream EBCDIC Convert RAM (BDP 3.5).
11. A MAC Write T2 (16-ns pulse) activates. Because no incoming MAC Byte Selects are active, no write occurs.
12. Channel Timing Chain T5 increments MAC Byte Counter (MAC 3.8).

Ready In To IOU departs Data Ready Control for IOU.

13. MAC Byte 1 Select activates in MAC Byte Selector (MAC 3.8).
14. IOU sends Ready Out and next data byte to MAC and starts Channel Timing Delay Chain. Data byte departs Data Mux 6.
15. Steps 7 through 10 repeat.

16. A MAC Write T2 gates A Write Byte 1.
17. MAC Data Bus Bits 0 through 7/8 enter RAM as EBCDIC BITS 8 through 15/33.
18. Channel Timing Chain T5 increments MAC Byte Counter (MAC 3.8).  
Data Ready Control sends Ready In To IOU to IOU.
19. MAC Byte 2 Select activates in MAC Byte Selector (MAC 3.8).
20. Steps 14 through 19 repeat with variations to select new byte until third byte enters RAM.
21. Advance activates in Address Increment Translator (MAC 3.6).  
Channel Timing Chain T5 increments MAC Byte Counter (MAC 3.8).  
Data Ready Control sends Ready In To IOU to IOU.
22. MAC Address Bits 0 through 7/8 increment in Address Incrementer (MAC 3.6).  
MAC Byte 0 Select activates in MAC Byte Selector (MAC 3.8).
23. Step 14 repeats.
24. Three bytes enter next address in A Stream EBCDIC Convert RAM in manner described above. At conclusion, Incremented MAC Address Bits address next RAM location. MAC Byte Counter returns to zero.  
  
Loading of A Stream EBCDIC Convert RAM continues until MAC Address Bit 0 switches from 1 to 0 (count equals 257).  
  
Address Increment Bits 8, 9 (MAC 3.2) are duplicates of MAC Address Bits 0, 1. When MAC Address Bit 2 toggles as count advances in Address Incrementer (MAC 3.6), Increment advances the count in the five-bit Partial Address Incrementer (MAC 3.2). When Address Increment Bit 8 switches from 1 to 0 (count equals 257), Address Increment Bits 5 through 7 (MAC 3.2) increment.  
  
During next byte transfer, BDP Go Select 1 activates in BDP Control Memory Go Decoder (MAC 3.2). Refer to step 7. Loading of B Stream EBCDIC Convert RAM (BDP 3.8) begins and follows pattern of writing into A Stream EBCDIC Convert RAM.  
  
Four other RAMs in BDP are loaded in sequence, following the pattern of writing in A Stream EBCDIC Convert RAM. BDP Go Select 2 enables loading of EBCDIC Encode RAM (BDP 3.22). Only bytes 2 and 3 of each address are loaded.  
  
BDP Go Select 3 enables loading of Convert To Binary/Decimal RAM (BDP 3.23). Only bytes 1 through 3 of each address are loaded. MAC Off Line selects MAC address.  
  
BDP Go Select 4 enables loading of Instruction Specification Error RAM (BDP 3.3). Only bytes 2 and 3 of each address are loaded. MAC Off Line selects MAC Address.  
  
BDP Go Select 5 enables loading of Convert Multiply By 256 RAM (BDP 3.27). Only bytes 1 through 3 of each address are loaded. MAC Off Line selects MAC Address.
25. IOU terminates operation.



Figure 2-4 shows address fields and various data formats used in writing BDP memories.

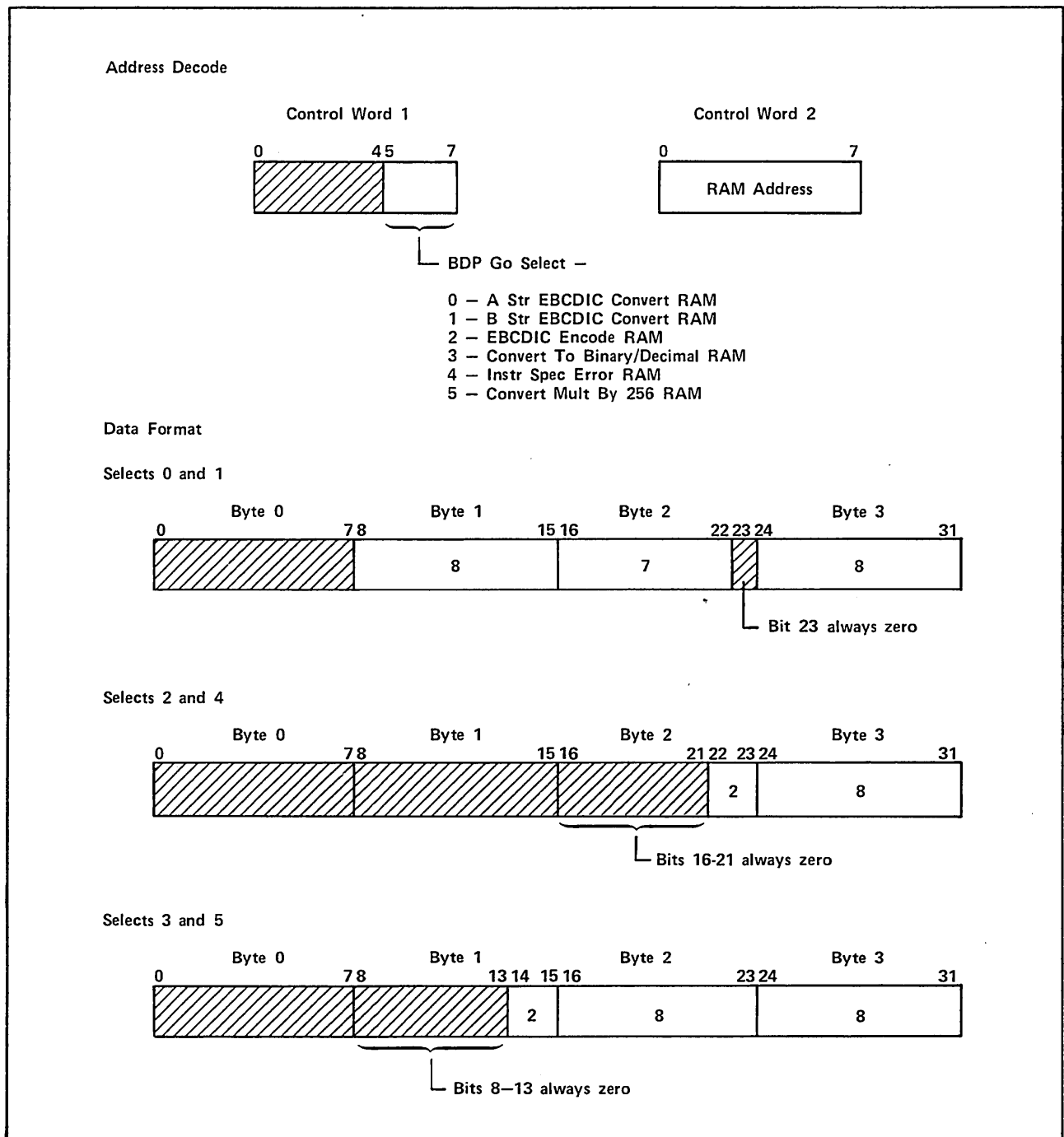


Figure 2-4. Type Code 5 - BDP Memories Address, Data Formats

#### First Level Instruction Decoder A, B Write (Function Code = 56)

The First Level Instruction Decoder (IF 3.1) provides RAM storage for 256 C170 instruction codes and 256 C180 instruction codes. Inactive bit 7 in the Partial Address Incrementer (MAC 3.2) and the address contained in the Address Incrementer (MAC 3.6) control writing of the C180 Decode RAMs (IF 3.1). When bit 7 in the Partial Address Incrementer toggles at the end of the C180 write operation, a similar write sequence occurs with C170 Decode RAMs (IF 3.1).

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations. Refer to figure 2-5 for contents of Control Words 1 and 2.

Opcode of five activates Operation Decode Equals 5 in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Type Code of six activates Gated Type Decode Equals 6 and Memory Type 3, 4, 5, or 6.

Operation Decode Equals 5 enters Read and Write FFs translator (MAC 3.4) to partially enable Set Write. Inactive Gate Type Decode Equals 0 activates Direct.

Gated Type Decode Equals 6 partially enables MAC Write Enable and Go IF in IF Decode RAM Interface (MAC 3.4). Memory Type 3, 4, 5, or 6 partially enables Advance in Address Increment Translator (MAC 3.6).

Control Word 1, containing Enable Load C170 RAM in bit 7 position, enters Partial Address Incrementer (MAC 3.2) as Maintenance Channel Data Bits 5 through 7. Control Word 2, containing RAM address, enters Address Incrementer (MAC 3.6) as Maintenance Channel Data Bits 0 through 7/8. Bits 0, 1 of RAM address also enter Partial Address Incrementer as Bits 8, 9. MAC Address Bits 0 through 7/8 depart Address Incrementer on MAC Address Bus (MAC 3.6) for Instruction Assembly (IF 3.0). Inactive Enable Load C170 RAMs and MAC Off Line generate MAC to RAM Select 1 in MAC to RAM Selector (IF 3.0). MAC Address Bits 0 through 7/8 become bits 0 through 7/64 in Instruction Assembly Mux (IF 3.0).

2. IOU sends Ready Out and data byte to MAC. Refer to IOU Read, Write Operations, Ready Out description.

Data byte enters Maintenance Channel Data Parity Check (MAC 3.0) as Maintenance Channel Data Bits 0 through 7/8 and proceeds to Data Mux 6 (MAC 3.0).

3. MAC Data Bus Bits 0 through 7/9 arrive at First Level Instruction Decoder A, B (IF 3.1) from Data Mux 6.
4. Ready 1 generates Set Write in Read and Write FFs translator (MAC 3.4) and Set Write Operation in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2).
5. Channel Timing Delay Chain starts (MAC 3.3).

Write FF sets (MAC 3.4).

6. Write Operation from Control FF (MAC 3.2) enables decode of MAC Byte Counter bits (MAC 3.8).

Read or Write partially enables Go IF, and Write partially enables MAC Enable Write in IF Decode RAM Interface (MAC 3.4).

Read or Write also partially enables Increment in Channel Byte Counter, Decoder (MAC 3.3) and Increment Counter in MAC Byte Counter translator (MAC 3.8).

7. At Channel Timing Chain T3, Go IF activates in IF Decode RAM Interface (MAC 3.4).
8. Go IF and MAC Off Line generates MAC Go in MAC Mode Control (IF 3.0).
9. MAC Go activates Enable Bytes 0, 1.
10. MAC address enters Instruction Assembly Register (IF 3.0) and departs as Instruction Assembly Bits 0 through 7/64.
11. Instruction Assembly Bits 0 through 7 emerge from C180 Parcel Select Mux (IF 3.0) as C180 Opcode Bits 0 through 7 to address C180 RAM A, B (IF 3.1).
12. Channel Timing Chain T4 activates.  
  
Set Ready In To IOU becomes active in Data Ready Control translator (MAC 3.3).
13. MAC Write Enable activates in IF Decode RAM Interface (MAC 3.4).
14. MAC Write Enable, MAC Off Line and inactive Enable Load C170 RAMs gate IF Byte Count Bits 0, 1 to C180 Write Enable Decoder (IF 3.0).
15. C180 A Write Enable activates and loads MAC Data Bus Bits 3 through 7/9 into byte 0 location (in C180 RAM A).
16. Channel Timing Chain T5 increments Channel Byte Counter (MAC 3.3) and MAC Byte Counter (MAC 3.8).  
  
Data Ready Control sends Ready In To IOU to IOU.
17. IOU sends Ready Out and next data byte to MAC and starts Channel Timing Delay Chain.
18. Steps 7 through 16 repeat. (Variation, step 15 - IF Byte Count decode of one activates C180 B Write Enable. MAC Data Bus Bits 3 through 7/9 enters byte 1 location (in C180 RAM B).)
19. Step 17 repeats, followed by steps 7 through 15. (Variation, step 15 - because IF Byte Count equals two, no data enters C180 RAM A or B (IF 3.1).)
20. Step 17 repeats, followed by steps 7 through 16 (Variation, step 15 - because IF Byte Count equals three, no data enters C180 RAM A or B.)
21. Channel Timing Chain T5 and MAC Byte Count Equals 3 or 7 activate Advance in Address Increment Translator (MAC 3.6).  
  
Channel Timing Chain T5 increments MAC and Channel Byte counters.  
  
Data Ready Control sends Ready In To IOU to IOU.
22. MAC Address Bits 0 through 7/8 increment in Address Incrementer (MAC 3.6).
23. Step 17 repeats.  
  
Two bytes enter next address in C180 RAM A, B in manner described above. At conclusion, incremented MAC Address Bits address next RAM location (MAC Byte Count is now seven) and MAC and Channel Byte counters return to zero.

Loading of C180 RAM A, B continues according to this pattern until MAC Address Bit 0 Switches from 1 to 0 (count equals 257) and Address Increment Bit 7 (MAC 3.2) sets. Refer to BDP Memory Write, step 24, for explanation of relationship between MAC Address Bit 0 and Address Increment Bit 7.

Address Increment Bit 7 enters IF Decode RAM Interface (MAC 3.4), where it becomes Enable Load C170 RAMs. Enable Load C170 RAMs gates IF Byte Count Bits 0, 1 to C170 Write Enable Decoder (IF 3.0). Loading of C170 Odd and Even Address RAMs proceeds in the same manner as for C180 RAM A, B except that all four RAMs associated with each address receive data byte. C170 Write Enable Decoder determines order of byte entry. MAC Data Bus Bits 2 through 7/9 enter C170 RAMs.

MAC To RAM Select 2 is active in MAC To RAM Selector (IF 3.0) during C170 RAM A, B write. It gates MAC Address Bits 0 through 7 to Instruction Assembly Register (IF 3.0) in bit positions 4 through 11. When Instruction Assembly Bits 4 through 11 enter C170 Parcel Select Mux (IF 3.0), they proceed to First Level Instruction Decoder A, B (IF 3.1) as C170 Opcode Bits 0 through 7 to provide address for loading of RAMs.

#### 24. IOU terminates operation.

Figure 2-5 shows address fields and data formats used in writing First Level Instruction Decoder RAMs.

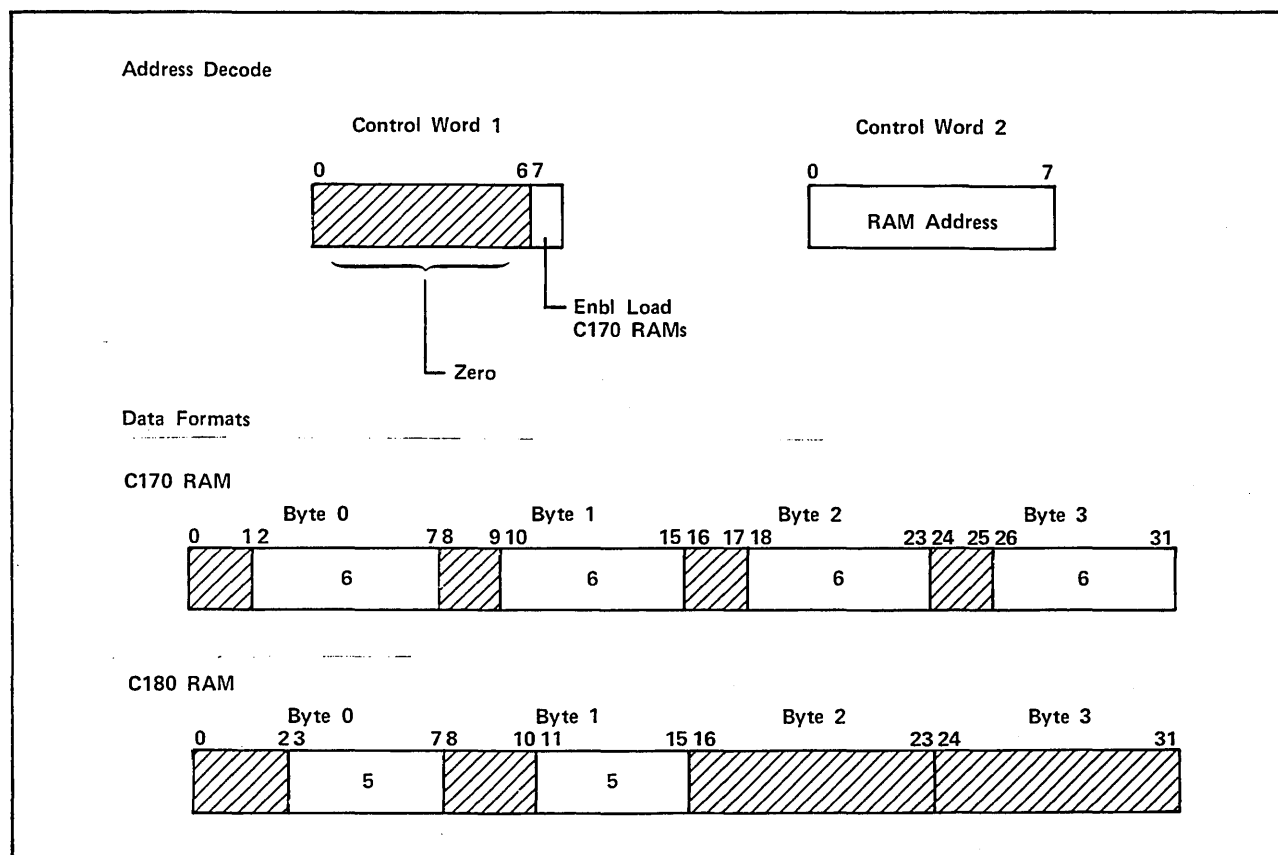


Figure 2-5. Type Code 6 - First Level Instruction Decoder Address, Data Formats

#### Control Memory Read (Function Codes = 43 through 46)

Control Memory read data is not directly available to MAC. Verification of Control Memory data is accomplished by parity checking at the RAM outputs during Type 3, 4, 5, and 6 operations. The Control Memory address returns to MAC as read data during all Control Memory read operations.

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations.

Opcode of four activates Operation Decode Equals 4 in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Appropriate type decode signals also activate.

Memory Type 3, 4, 5, or 6 partially enables Advance in Address Increment Translator (MAC 3.6). Gated Memory Type 3, 4, 5, or 6 activates Mux 3 Select 1 in Data Mux 3 Select Control (MAC 3.1). Mux 3 Select 1 selects MAC Address Bits 0 through 7/8 in Data Mux 3 (MAC 3.1).

Operation Decode Equals 4 enters Read and Write FFs translator (MAC 3.4) to partially enable Set Read. Inactive Gated Type Decode Equals 0 activates Direct.

Control Words 1 and 2 (as required by specific operation) enter Address Incrementer (MAC 3.6) and Partial Address Incrementer (MAC 3.2). Refer to step 1 of each Control Memory write description.

Second Activate Out generates Set Read in Read and Write FFs translator. Set Read Operation activates in Maintenance Channel Opcode, Type Code Register, Decoder.

2. Read FF sets (MAC 3.4).

Channel Timing Delay Chain (MAC 3.3) starts.

Set Read deactivates.

Set Read Operation enters Control FF in Maintenance Channel Opcode, Type Code Register, Decoder.

3. Read or Write partially enables Set Ready In To IOU in Data Ready Control translator (MAC 3.3).

Read Operation enters Address Increment Translator (MAC 3.6) and partially enables Advance.

4. At Channel Timing Chain T4, Set Ready In To IOU activates.

Gate Channel Output Register activates.

5. Gate Channel Output Register loads MAC Address Bits 0 through 7/8 into Channel Output Register (MAC 3.1). MAC Address originates in Address Incrementer (MAC 3.6) and reaches Channel Output Register by way of MAC Address Bus.

6. Channel Timing Chain T5 activates, generating Advance in Address Increment Translator (MAC 3.6).

Data Ready Control sends Ready In To IOU to IOU.

7. MAC Address Bits 0 through 7/8 increment in Address Incrementer (MAC 3.6).

8. IOU sends Ready Out to MAC (refer to IOU Read, Write Operations, Ready Out description for details) and starts Channel Timing Delay Chain.
9. Steps 4 through 8 repeat.

Operation continues in manner described above until all addresses in all RAMs associated with operation have been swept.

MAC Address Bits 0 through 7/8 enter Reference ROM, Soft Control Memories, BDP Memories, and First Level Instruction Decoder in the same way they enter during write operations. Refer to specific Control Memory write description for details.

### Register File Read, Write

The 64-word Register File provides storage for operating registers (A, B, and X) for C170 and C180 instructions. It also holds other exchange package data and intermediate results. The IOU has direct read and write access to the Register File by way of the MAC Address and Data buses and the CST Disassembly Network (CST 3.5).

Control Word 2 provides the read or write address to the Register File through the MAC Address Bus and Register File Address Selector (OPI 3.3).

Transfers may exceed eight bytes. The Register File address increments once for every eight bytes transferred.

MAC must be operating off line. No Start operation may have occurred to place the CP on line.

### Register File Read (Function Code = 47)

Refer to figure 2-1 and accompanying MAC Operation text for details about Register File read data paths.

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations.

Opcode of four activates Operation Decode Equals 4 in Maintenance Channel Opcode, Type Code Register Decoder (MAC 3.2). Type Code of seven activates Type Decode Equals 7.

Operation Decode Equals 4 enters Read and Write FFs translator (MAC 3.4) to partially enable Set Read.

Type Decode Equals 7 disables Address Equals 1X and Not Type 7 (MAC 3.5), which causes Mux 5 Select 1 to activate in Data Mux 4, 5 Select Translator (MAC 3.0). MAC Data Bus Bits 0 through 7/8 are selected in Data Mux 5 (MAC 3.0).

Type Decode Equals 7 partially enables Advance in Address Increment Translator (MAC 3.6) and Register File Go in Register File Go Control (MAC 3.8). Type Decode Equals 1 or 7 deactivates Select PFS Bus in CST Disassembly Control (MAC 3.8), which in turn enables CST Byte Count Bit 0 to CST Disassembly Network (CST 3.5).

Inactive Gated Type Decode Equals 0 activates Direct in Read and Write FFs translator (MAC 3.4). Inactive Type Decode Equals 1 activates Select Register File to Disassembly Network in CST Disassembly Control (MAC 3.8).

Control Word 2, containing Register File (OPI 3.5) read address, enters Address Incrementer (MAC 3.6). MAC Address Bits 0 through 5 proceed to Register File Address Selector (OPI 3.3). MAC Off Line gates bits 0 through 5 to Register File Address Register (OPI 3.5) as Register File Address Bits 0 through 5.

Register File Read Data Bits 0 through 63/64 through 71 (OPI 3.5) enter CST Disassembly Network (CST 3.5) and are gated to Disassembly Register input by Select Register File to Disassembly Network.

Second Activate Out generates Set Read Operation in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2) and Set Read in Read and Write FFs translator (MAC 3.4).

2. Read FF Sets (MAC 3.4).

Channel Timing Delay Chain (MAC 3.3) starts.

Set Read deactivates.

Read Operation activates in Control FF (MAC 3.2).

3. Read or Write partially enables Increment Counter in MAC Byte Counter translator (MAC 3.8).

Read Operation activates Register File Go in Register File Go Control (MAC 3.8) for duration of function.

4. Register File Go leaves Holding Register (MAC 3.8) and enters CST Disassembly Network Interface (OPI 3.4). Register File Go generates Strobe Disassembly Register, a continuous input to CST Disassembly Register (CST 3.5).

5. Register File Read Data Bits 0 through 63/64 through 71 enter Disassembly Register.

6. CST Byte Count Bits 0 through 2 equal to zero select high-order byte in Disassembly Muxes 1, 2 for output to Data Mux 5 (MAC 3.0) as MAC Bus Data Bits 0 through 7/8.

7. Data Mux 5 Bits 0 through 7/8 enter Data Mux 3 (MAC 3.1) and proceed to Channel Output Register.

8. At Channel Timing Chain T4, Set Ready In To IOU activates in Data Ready Control translator (MAC 3.3).

9. Gate Channel Output Register loads Channel Output Register (MAC 3.1).

10. Ready In To IOU proceeds from Holding Register (MAC 3.3) to IOU, accompanying data byte.

Channel Timing Chain T5 increments MAC Byte Counter (MAC 3.3). MAC Byte Counter Bits 0 through 2 depart CST Disassembly Control (MAC 3.8) as CST Byte Count.

11. IOU sends Ready Out to MAC (refer to IOU Read, Write Operations, Ready Out description for details) and starts Channel Timing Delay Chain.

12. Steps 8 to 11 repeat. Incremented CST Byte Count bits select new byte each cycle in CST Disassembly Network (CST 3.5) until eight bytes depart MAC for IOU.

13. When Channel Timing Chain T5 activates during eighth byte transfer (step 10), Advance increments MAC Address Bits 0 through 7/8 in Address Incrementer (MAC 3.6).

Channel Timing Chain T5 presets MAC Byte Counter to zero.

14. Steps 5 through 7 and 11 repeat.
15. Steps 8 through 14 repeat until IOU terminates operation.

#### Register File Write (Function Code = 57)

Refer to figure 2-1 and accompanying MAC Operation text for details about Register File write data paths.

1. Function Code and Control Words 1, and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations.

Opcode of five activates Operation Decode Equals 5 in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Type Code of seven activates Type Decode Equals 7.

Operation Decode Equals 5 enters Read and Write FFs translator (MAC 3.4) to partially enable Set Write. Operation Decode Equals 5 also partially enables Set Write Operation at Output of Opcode Decoder 2 (MAC 3.2).

Type Decode Equals 7 partially enables Advance in Address Increment Translator (MAC 3.6) and Register File Go in Register File Go Control (MAC 3.8). Type Decode Equals 7 also partially enables Decoder Enable in Register File Byte Selector (MAC 3.8).

Inactive Gated Type Decode Equals 9 activates Direct in Read and Write FFs translator (MAC 3.4).

Control Word 2, containing Register File (OPI 3.5) write address, enters Address Incrementer (MAC 3.6). MAC Address Bits 0 through 5 proceed to Register File Address Selector (OPI 3.3). MAC Off Line gates bits 0 through 5 to Register File Address Register (OPI 3.5) as Register File Address Bits 0 through 5.

2. IOU sends Ready Out and data byte to MAC. Refer to IOU Read, Write Operations, Ready Out description for details.

Data byte enters Maintenance Channel Data Parity Check (MAC 3.0) as Maintenance Channel Data Bits 0 through 7/8 and proceeds to Data Mux 6 (MAC 3.0). Bit 8 of even byte enters MAC Even Byte Parity Register.

3. MAC Data Bus Bits 0 through 7/8, 9 depart Data Mux 6 for Register File Write Data Mux (OPI 3.5).

4. Select MAC Data, activated by MAC Off Line in CST Disassembly Network Interface (OPI 3.4), gates data byte to Register File RAM (OPI 3.5).

5. Ready 1 generates Set Write in Read and Write FFs translator (MAC 3.4) and Set Write Operation in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2).

6. Channel Timing Delay Chain starts (MAC 3.3) and Write FF sets (MAC 3.4).

Write Operation goes to Register File Byte Selector (MAC 3.8) from Control FF (MAC 3.2).

7. Read or Write partially enables Increment Counter in MAC Byte Counter (MAC 3.8) translator.

Write Operation activates Decoder Enable in Register File Byte Selector.



8. MAC Byte Count Bits 0 through 2 equal to zero activate Register File Byte 0, 1 Select in Register File Byte Selector.
9. Register File Byte 0, 1 Select enter Register File Write Control (OPI 3.4) where they AND with Select MAC Data.
10. Channel Timing Chain T2 activates Register File Go in Register File Go Control (MAC 3.8).
11. Write Register File gates Register File Byte 0, 1 Select to input of Holding Register in Register File Write Control (OPI 3.4).
12. Register File Byte 0, 1 Write Enable depart Holding Register and gate MAC Data Bus Bits 0 through 7/8, 9 into addressed Register File RAM location as Write Data Bits 0 through 7, 8 through 15/64, 65.
13. At Channel Timing Chain T4, Set Ready In To IOU activates in Data Ready Control translator (MAC 3.3).
14. At Channel Timing Chain T5, Increment Counter increments MAC Byte Count Bits 0 through 2 in MAC Byte Counter (MAC 3.8).

Data Ready Control sends Ready In To IOU to IOU.

15. Register File Byte 0 Select deactivates in Register File Byte Selector (MAC 3.8).
16. IOU sends Ready Out and next data byte to MAC and starts Channel Timing Delay Chain.
17. Steps 10 through 13 repeat. (Variation, step 11 - only Register File Byte 1 Select enters Register File Write Control (OPI 3.4).)

Register File Byte 1 Select gates MAC Data Bus Bits 0 through 7/8, 9 (byte 1) into addressed Register File RAM location as Write Data Bits 8 through 15/64, 65.

Bit 8 is previous byte's parity, latched in MAC Even Byte Parity Register (MAC 3.0). Even byte locations in RAM receive no parity. Parity for odd byte and saved parity for even byte are stored with odd byte. Refer to Soft Control Memory Write, step 16, for further explanation.

18. Channel Timing Chain T5 increments MAC Byte Counter (MAC 3.8).

Ready In To IOU departs Data Ready Control for IOU.

19. Register File Byte 2, 3 Select activate in Register File Byte Selector (MAC 3.8).
20. Step 16 repeats.

21. Steps 10 through 19 repeat. (Variation - byte 2 enters byte 2 and 3 locations, after which byte 3 enters byte 3 location. Once both locations have been filled, Register File Bytes 4, 5 Select activate and data entry into these positions proceeds in manner of preceding bytes. Loading of byte positions 6 and 7 follows.)

When Channel Timing Chain T5 activates during eighth byte transfer (MAC Byte Count Equals 7), Advance increments MAC Address Bits 0 through 7/8 in Address Incrementer (MAC 3.6). MAC Byte Counter returns to zero.

With arrival of next data byte from IOU, steps 10 through 21 repeat at next Register File RAM address. Writing of Register File RAM continues until IOU terminates operation.

## MAC READS, WRITES

MAC reads and writes comprise a subcategory of IOU-initiated direct operations. Data transfer is to or from Type 0 registers that reside in MAC or PMF. Control Word 2 provides the specific register's address. Table 2-1 lists access restrictions.

MAC reads and writes use MAC conflict-control circuitry to reserve the MAC Data Bus because the CP also may enter the registers MAC is accessing. The conflict-control network permits the IOU to transfer data on the MAC Data Bus, subject to interruption by a CP request between byte transfers. The IOU cannot interrupt CP data transfers while the CP is reading or writing these registers.

Registers residing in MAC which are subject to MAC reads and writes include PFS, DEC, PTM, Element ID, Processor ID, and Options Installed registers. Keypoint and Control registers in PMF also are subject to MAC reads and writes.

### Element ID, Processor ID, Options Installed Read (Function Code = 40, Address = 10, 11, 12)

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations.

Opcode of four activates Operation Decode Equals 4 in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Type Code of zero activates Gated Type Decode Equals 0.

Gated Type Decode Equals 0 disables Direct in Read and Write FFs translator (MAC 3.4).

Control Word 2 enters Address Incrementer (MAC 3.6). MAC Address Bits 0 through 7/8 enter MAC Address Decoder (MAC 3.5). Microcode Not Needed, Address Equals 1X and Not Type 7, and appropriate Lower Decoder output activate.

If address equals 10 (Element ID), Mux 5 Select 2 activates in Data Mux 4, 5 Select Translator (MAC 3.0). If address equals 11 (Processor ID), Mux 5 Select 1 and 2 activate, as do Mux 4 Select 1 and 2. If address equals 12 (Options Installed), Mux 4 Select 2 and Mux 5 Select 1 and 2 activate.

Because MAC Byte Counter (MAC 3.8) is initially set to zero, Address Equals 1X and Not Type 7 activates Zero Fill in Maintenance Register Zero Fill Translator (MAC 3.0). Data Mux 5 Bits 0 through 7/8 are zero filled.

Second Activate Out generates Set Read Operation in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Set Read Operation, Microcode Not Needed, and inactive Direct generate Set MAC Read in Read, Write Control (MAC 3.6).

2. Read Operation activates in Control FF (MAC 3.2).

MAC Read FF sets in Read, Write Control (MAC 3.6).

3. MAC Read activates MAC Read or Write in Data Ready Control (MAC 3.3).

MAC Read or Write partially enables Set Ready In To IOU.

MAC Read or Write also enters MAC Busy Control (MAC 3.7) and generates MAC Read, Write, or Clear, provided MAC Busy 1, 2 are inactive. Active Busy 1 or 2 indicates CP is using shared data paths in MAC and prevents IOU from starting read or write sequence under control of MAC Timing Chain (MAC 3.7).

4. MAC Read, Write, or Clear activates Start MAC Timing Chain in Start MAC Timing Chain Translator (MAC 3.7).
5. MAC Busy 1 FF sets.  
MAC Read, Write, or Clear deactivates.
6. MAC Busy 1 enters Condition Sensing Mux (CST 3.2) to suspend execution of any Micrand needing to use shared MAC data paths.
7. MAC Busy 2 Register sets.
8. MAC Timing Chain T4 activates Ready In Enable in Disconnect Translator (MAC 3.5).
9. Ready In Enable, MAC Read or Write, and inactive MAC Timing Chain T5 generate Set Ready In To IOU in Data Ready Control (MAC 3.3).
10. Gate Channel Output Register loads zero-filled Data Mux 5 Bits 0 through 7/8 into Channel Output Register (MAC 3.1).
11. MAC Timing Chain T5 activates Byte Transfer or Clear Error Complete in Read, Write Control (MAC 3.6).

Increment Counter increments MAC Byte Count Bits 0 through 2 in MAC Byte Counter (MAC 3.8) and enables MAC data parity check output in PFS Translator (MAC 3.8).

Clear MAC Busy 1 activates in Clear MAC Operation Translator (MAC 3.7) because MAC Response/Request From Pipeline is active.

Ready In To IOU from Data Ready Control accompanies data byte to IOU.

12. MAC Read FF (MAC 3.6) clears.  
MAC Busy 1 FF clears, enabling CP to make MAC request.
13. MAC Busy 2 Register clears.
14. IOU sends Ready Out to MAC. Refer to IOU Read, Write Operations, Ready Out description for details.
15. Combination of Read Operation and Ready 1 activates Set MAC Read in Read, Write Control (MAC 3.6).
16. Steps 2 through 16 repeat until five zero-filled bytes have been sent to IOU and MAC Byte Counter increments to five. (Variation, step 2 - once Control FF sets (MAC 3.2) for first byte transfer, Read Operation remains active until new function enters MAC from IOU.)

If address equals 10 (Element ID), Zero Fill deactivates in Maintenance Register Zero Fill Translator after MAC Byte Count reaches five. Output of Data Mux 5 (MAC 3.0) is gated to Data Mux 3 (MAC 3.1). With Mux 5 Select 2 active, Element Identification Bits 40 through 47/Parity become Data Mux 5 Bits 0 through 7/8 for next byte transfer. After MAC Byte Counter increments to six, MAC/IU Byte Count Equals 0, 2, 4, or 6 activates Mux 4 Select 1 and Mux 5 Select 1. Element Identification Bits 48 through 55/Parity become next byte transferred to IOU. After MAC Byte Counter increments to seven, Mux 4 Select 1 deactivates. Element Identification Bits 56 through 63/Parity become last byte transferred to IOU.

If address equals 11 (Processor ID), Zero Fill remains active for two additional byte transfers. After MAC Byte Counter increments to seven, Zero Fill deactivates in Maintenance Register Zero Translator (MAC 3.0). Address translations gate Processor Identification, seven zero bits, and parity through Data Muxes 4 and 5 as eighth byte to IOU.

If address equals 12 (Options Installed), Zero Fill remains active for seven byte transfers. After MAC Byte Counter increments to seven, Zero Fill deactivates. Address translations gate four zero bits, Options Installed Bits 0 through 3, and parity through Data Muxes 4 and 5 as eighth byte to IOU.

IOU terminates operation.

#### DEC, PTM, PFS Register Read (Function Code = 40, Address = 30, A0, 8X)

Refer to figure 2-1 and accompanying MAC Operation text for details about DEC, PTM, and PFS Register read data paths.

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations.

Opcode of four activates Operation Decode Equals 4 in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Type Code of zero activates Type Decode Equals 0 and Gated Type Decode Equals 0.

Gated Type Decode Equals 0 disables Direct in Read and Write FFs translator (MAC 3.4). Type Decode Equals 0 activates Mode Select Enable in PFS Register Select Translator (MAC 3.5). Active Select PFS Bus disables CST Byte Count Bit 0 in CST Disassembly Control (MAC 3.8). PFS Bus Data Output Bits 0 through 7/8 are selected in CST Disassembly Network (CST 3.5).

Control Word 2 enters Address Incrementer (MAC 3.6). MAC Address Bits 0 through 7/8 enter MAC Address Decoder (MAC 3.5). Microcode Not Needed activates and partially enables Set MAC Read in Read, Write Control (MAC 3.6).

DEC, PTM, or PFS activates in PFS Register Select Translator (MAC 3.5). PFS and Address Bit 7 determine whether odd or even PFS register read is enabled.

DEC or PTM partially enables MAC/IU Byte Count path to DEC or PTM Board Select Decoder (MAC 3.5). PFS partially enables Address Bits 4 through 6 to PFS Board Select Decoder.

Mux 5 Select 1 is active in Data Mux 4, 5 Select Translator (MAC 3.0), gating MAC Data Bus Bits 0 through 7/8 from CST Disassembly Network into Data Mux 5 (MAC 3.0).

Second Activate Out generates Set Read Operation in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Set Read Operation, Microcode Not Needed, and inactive Direct activate Set MAC Read in Read, Write Control (MAC 3.6).

2. Read Operation activates in Control FF (MAC 3.2).

MAC Read FF sets in Read, Write Control.

3. MAC Read activates MAC Read or Write in Data Ready Control (MAC 3.3).

MAC Read or Write partially enables Set Ready In To IOU in Data Ready Control.

MAC Read or Write also enters MAC Busy Control (MAC 3.7) and generates MAC Read, Write or Clear, provided MAC Busy 1 and 2 are inactive. Refer to Read Element ID, Processor ID, Options Installed description, steps 2 through 7, for explanation of MAC Busy conflict-control network.

4. MAC Read, Write, or Clear activates Start MAC Timing Chain in Start MAC Timing Chain Translator (MAC 3.7).
5. MAC Timing Chain starts.  
  
If DEC read, refer to steps 6 through 9, 19 through 27; if PTM read, refer to steps 6, 10 through 27; if PFS read, refer to steps 28 through 38.
6. MAC Timing Chain T1 (and subsequently MAC Timing Chain T2-T4) gates MAC/IU Byte Count Bits 0 and 1 to DEC or PTM Board Select Decoder (MAC 3.5). MAC Byte Counter is set to zero initially. Bits 0 and 1 increment once for every two bytes transferred. When incremented, bits 0 and 1 select board containing next two DEC or PTM bytes to be transferred.
7. If DEC is active, DEC Zero Fill activates at output of DEC Board Select Decoder and continues for 256 ns.
8. DEC or PTM Zero Fill generates Zero Fill in Maintenance Register Zero Fill Translator (MAC 3.0).
9. Zero-filled Data Mux 5 Bits 0 through 7/8 proceed to Data Mux 3 and Channel Output Register (MAC 3.1).
10. If PTM is active, PTM Board Select 2 activates in PTM Board Select Decoder.
11. Board Select 2 activates (MAC 3.5) and remains active for 256 ns.
12. Board Select 2 enters Holding Register in Set/Clear PFS Control (MAC 3.10) and generates Board 2 Register Select Odd, Even.
13. Select DEC/PTM Register on Board 2 activates.
14. Board 2 Mux C Select 2 and 1 activate in PFS Bus Control Translator (MAC 3.10), because MAC/IU Byte Count equals zero. State of MAC/IU Byte Count Bit 2 dictates which byte of byte pair in Board 2 PTM Register enters mux.
15. Board 2 Mux C Select 2 and 1 select PTM Register Bits 0 through 7/16 in Board 2 Mux C (MAC 3.11).
16. PFS Bus Data Bits 0 through 7/8 proceed through Boards 3, 4 Mux C to enter CST Disassembly Network (CST 3.5) as PFS Bus Data Output Bits 0 through 7/8.
17. PFS Bus Data Output Bits 0 through 7/8 enter Disassembly Mux 2 (CST 3.5) and from there proceed to Data Mux 5 (MAC 3.0) as MAC Data Bus Bits 0 through 7/8.
18. Data byte passes through Data Mux 3 to Channel Output Register input (MAC 3.1).
19. At MAC Timing Chain T4, Ready In Enable activates in Disconnect Translator (MAC 3.3) and generates Set Ready In To IOU in Data Ready Control (MAC 3.3).
20. Gate Channel Output Register loads data byte into Channel Output Register.

21. MAC Timing Chain T5 activates Clear MAC Busy 1 in Clear MAC Operation Translator (MAC 3.7), and Byte Transfer or Clear Error Complete in Read, Write Control (MAC 3.6).

MAC Timing Chain T5 also increments MAC Byte Counter (MAC 3.8).

Ready In To IOU from Data Ready Control accompanies data byte to IOU.

22. MAC Read FF (MAC 3.6) clears. MAC Busy 1 FF clears, enabling CP to interrupt with MAC request.
23. MAC Busy 2 Register clears.
24. IOU sends Ready Out to MAC. Refer to IOU Read, Write Operations, Ready Out description for details.
25. Combination of Read Operation and Ready 1 activates Set MAC Read in Read, Write Control (MAC 3.6).
26. Steps 2 through 25 repeat with variations required to select new byte or zero-fill input until eight-byte transfer is complete. (Variation, step 2 - Control FF sets (MAC 3.2) for first byte transfer. Read Operation remains active until new function enters MAC from IOU.)

First four bytes of DEC transfer are zero-filled. Next two byte pairs are selected in Board 0 and 1 Mux C (MAC 3.11), respectively. First three byte pairs of PTM transfer are selected in Board 2, 3, and 4 Mux C, respectively. Last two bytes of PTM transfer are zero-filled.

27. IOU terminates operation.
28. MAC Timing Chain T1 (and subsequently MAC Timing Chain T2 through T4) gates Address Bits 4 through 6 to PFS Board Select Decoder (MAC 3.5). Bits 4 through 6 specify board containing odd and even PFS register pair, for example, registers 80, 81, one of which is to be read.
29. PFS Board Select N enters Board Select Register (MAC 3.5) to become Board Select N. PFS Board Select N remains active 256 ns.
30. Board Select N enters Holding Register in Set/Clear PFS Control (MAC 3.10) and generates Board N Register Select Odd or Board N Register Select Even. Address translation in PFS Register Select Translator (MAC 3.5) determines which one.
31. Board N Odd Register Select or Board N Even Register Select activates in Set/Clear PFS Control translator.

Byte selection takes place as follows: Because IU/MAC Byte Count initially equals zero, bits 0 and 1 of byte count select two high-order bytes of all five even PFS registers in PFS Mux A (MAC 3.10).

Inactive MAC/IU Byte Count Bit 2 enables byte 0 of all 10 PFS registers.

Register Select Odd, based on address translation in PFS Register Select Translator, determines whether five odd or five even PFS register byte 0's reach Buffer Register input (MAC 3.10).

Board Select N determines which byte 0 enters Buffer Register.

32. Board N Mux C Select 1 activates in PFS Bus Control Translator (MAC 3.10) to place selected byte on PFS Data Bus.
33. PFS Bus Data Bits 0 through 7/8 depart Board N Mux C (MAC 3.11) for CST Disassembly Network (CST 3.5) as PFS Bus Data Output Bits 0 through 7/8.
34. PFS Bus Data Output Bits 0 through 7/8 enter Disassembly Mux 2 (CST 3.5) and from there proceed to Data Mux 5 (MAC 3.0) as MAC Data Bus Bits 0 through 7/8.
35. Data byte passes through Data Mux 3 to Channel Output Register input (MAC 3.1).
36. Refer to steps 19 to 25.
37. Steps 2 through 5, 28 through 36 repeat with variations required to select new byte or zero-fill until eight-byte transfer is complete. (Variation, step 2 - Control FF (MAC 3.2) sets for first byte transfer. Read Operation remains active until new function enters MAC from IOU.)

When MAC/IU Byte Count increments to even value, PFS Mux A (MAC 3.10) selects next two bytes from each even PFS register. MAC/IU Byte Count Bit 2 selects these two bytes consecutively in PFS Mux B.

Because odd PFS registers are two bytes long, last six bytes of odd PFS register transfer to IOU are zero-filled. PFS Odd and MAC/IU Byte Count Equals 2 through 7 generate Zero Fill in Maintenance Register Zero Fill Translator (MAC 3.0).

38. IOU terminates operation.

#### DEC, PTM, PFS Register Write (Function Code = 50, Address = 30, AO, 8X)

Refer to figure 2-1 and accompanying MAC Operation text for details about DEC, PTM, and PFS Register write data paths.

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations.

Opcode of five activates Operation Decode Equals 5 in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Type Code of zero activates Type Decode Equals 0 and Gated Type Decode Equals 0.

Operation Decode Equals 5 partially enables Set Write Operation in Maintenance Channel Opcode, Type Code Register, Decoder.

Gated Type Decode Equals 0 disables Direct in Read and Write FFs translator (MAC 3.4). Type Decode Equals 0 activates Mode Select Enable in PFS Register Select Translator (MAC 3.5).

Control Word 2 enters Address Incrementer (MAC 3.6). MAC Address Bits 0 through 7/8 enter MAC Address Decoder (MAC 3.5). Microcode Not Needed activates and partially enables Set MAC Write in Read, Write Control (MAC 3.6) translator.

DEC, PFS, or PTM activates in PFS Register Select Translator. PFS and Address Bit 7 determine whether odd or even PFS register write is enabled.

DEC or PTM partially enables MAC/IU Byte Count path to DEC or PTM Board Select Decoder (MAC 3.5). PFS partially enables Address Bits 4 through 6 to PFS Board Select Decoder.

2. IOU sends Ready Out and data byte to MAC. Refer to IOU Read, Write Operations, Ready Out description for details.

Data byte enters Maintenance Channel Data Parity Check (MAC 3.0) as Maintenance Channel Data Bits 0 through 7/8 and proceeds to Data Mux 6 (MAC 3.0).

3. MAC Data Bus Bits 0 through 7/9 depart for PFS Bus (MAC 3.11) by way of Register File Write Data Mux (OPI 3.5).
4. Ready 1 activates Set Write Operation in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2).
5. Set Write Operation activates Set MAC Write in Read, Write Control translator (MAC 3.6).
6. MAC Write FF sets in Read, Write Control.
7. MAC Write activates MAC Read or Write in Data Ready Control (MAC 3.3).

MAC Write also becomes Write Mode in PFS Register Select Translator (MAC 3.5).

8. MAC Read or Write partially enables Set Ready In To IOU in Data Ready Control.

MAC Read or Write also enters MAC Busy Control (MAC 3.7) and generates MAC Read, Write, or Clear, provided MAC Busy 1 and 2 are inactive. Refer to Read Element ID, Processor ID, Options Installed description, steps 2 through 7, for explanation of MAC Busy conflict-control network.

9. MAC Read, Write, or Clear activates Start MAC Timing Chain in Start MAC Timing Chain Translator (MAC 3.7).
10. MAC Timing Chain starts.

If DEC write, refer to steps 11, 12, 20 through 26; if PTM write, refer to steps 11, 13 through 26; if PFS write, refer to steps 27 through 35.

11. MAC Timing Chain T1 (and subsequently MAC Timing Chain T2 through T4) enables MAC/IU Byte Count Bits 0 and 1 to DEC or PTM Board Select Decoder (MAC 3.5). MAC Byte Count is set to zero initially. Bits 0 and 1 increment once for every two bytes transferred. When incremented, bits 0 and 1 select DEC or PTM Board that is to receive next two bytes.
12. If DEC is active, no DEC Board Select 1 signal is generated and no DEC register write occurs.
13. If PTM is active, PTM Board Select 2 activates in PTM Board Select Decoder.
14. Board Select 2 activates and remains active for 256 ns.
15. Board Select 2 enters Holding Register in Set/Clear PFS Control (MAC 3.10) and generates Board 2 Register Select Odd, Even, as well as Board 2 Write.
16. Board 2 Register Select Odd, Even activate Select DEC/PTM Register on Board 2.
17. Select DEC/PTM Register on Board 2, Board 2 Write and inactive MAC/IU Byte Count Bit 2 activate Board 2 PTM Write 0 in Write DEC/PTM Byte Select (MAC 3.10).
18. Board 2 PTM Write 0 gates PFS Bus Data Bits 0 through 7/8 to low-order byte position of Board 2 PTM Register (MAC 3.11).



19. Clock and Board Select 2 load data byte into PTM register.
20. Whether write has occurred (in case in PTM active) or not (in case of DEC active), MAC Timing Chain T4 activates Ready In Enable in Disconnect Translator (MAC 3.3).
21. Ready In Enable generates Set Ready In To IOU in Data Ready Control translator. (MAC 3.3).
22. MAC Timing Chain T5 generates Byte Transfer or Clear Error Complete in Read, Write Control translator (MAC 3.6).  
  
MAC Timing Chain T5 also increments MAC Byte Count (MAC 3.8) and activates Clear MAC Busy 1 in Clear MAC Operation Translator (MAC 3.7).  
  
Ready In To IOU proceeds to IOU from Data Ready Control.
23. MAC Write FF clears.  
  
MAC Busy 1 FF clears, enabling CP to interrupt with MAC request.
24. MAC Busy 2 Register clears.
25. Steps 2 through 24 repeat with variations required to write new byte or increment byte count without writing, until eight-byte transfer is complete.  
  
First four bytes of DEC transfer are not written into DEC registers. Next two byte pairs are written into Board 0 and 1 DEC registers, respectively. First three byte pairs of PTM write are written into Board 2, 3 and 4 PTM registers, respectively. Last two bytes from IOU are not written into PTM registers.
26. IOU terminates operation.
27. MAC Timing Chain T1 (and subsequently MAC Timing Chain T2 through T4) gates Address Bits 4 through 6 to PFS Board Select Decoder (MAC 3.5). Bits 4 through 6 specify board containing odd and even PFS register pair, for example, registers 80, 81, one of which is to be written.
28. PFS Board Select N enters Board Select Register (MAC 3.5) to become Board Select N. It remains active 256 ns.
29. Board Select N enters Holding Register in Set/Clear PFS Control (MAC 3.10) and generates Board N Register Select Even or Board N Register Select Odd. Translation of Address Bit 7 in PFS Register Select Translator (MAC 3.5) determines which one. Board N Write also activates.
30. Combination Board N Odd Register Select or Board N Even Register Select, Board N Write, and Board N PFS Bus Data Input Bit 7 sets selected PFS register. If Board N PFS Bus Data Input Bit 7 is clear, selected PFS register clears. Bit 7 is supplied to PFS Bus by IOU in Step 2 via MAC Data Bus.
31. MAC Timing Chain T5 generates Byte Transfer or Clear Error Complete in Read, Write Control translator (MAC 3.6) and activates Clear MAC Busy 1 in Clear MAC Operation Translator (MAC 3.7).
32. MAC Write FF clears.  
  
MAC Busy 1 FF clears, enabling CP to make MAC request.
33. MAC Busy 2 Register clears.

34. Steps 2 through 10, 27 through 33 repeat with bit 7 of each byte setting or clearing addressed PFS register until eight-byte transfer is complete. Bit 7 of last byte determines final state of selected PFS Register.
35. IOU terminates operation.

PMF Register 21 and 22 Read (Function Code = 40, Address = 21, 22)

PMF contains two registers which the IOU may read. Register 21 holds up to 16 64-bit words collected in conjunction with Keypoint instructions. The IOU reads up to 16 words in a single operation (reading buffer until empty).

Register 22 contains 48 bytes which control PMF operation, indicate PMF status, or are the output of eight 32-bit event/state counters. The IOU reads Register 22 at the end of a PMF operation. The IOU initiates PMF operations with a Register 22 write.

The data path to the IOU for both read operations is from the Read Output Mux (PMF 3.5) by way of Data Muxes 2 and 3 (MAC 3.1).

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations.

Opcode of four activates Operation Decode Equals 4 in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Type Code of zero activates Gated Type Decode Equals 0.

Gated Type Decode Equals 0 disables Direct in Read and Write FFs translator (MAC 3.4) and partially enables Mux 2 Select 1 in Data Mux 2 Select Control (MAC 3.1).

Control Word 2 enters Address Incrementer (MAC 3.6). MAC Address Bits 0 through 7/8 enter MAC Address Decoder (MAC 3.5). Microcode Not Needed activates and partially enables Set MAC Read in Read, Write Control (MAC 3.6). MAC Address Equals 2X and MAC Address Equals X1 or X2 partially enable PMF Transfer in PMF Interface Control (MAC 3.7). Mux 2 Select 1 activates in Data Mux 2 Select Control (MAC 3.1). Mux 3 Select 2 activates in Data Mux 3 Select Control (MAC 3.1).

If MAC Address is 22, Initialization Byte Counter (PMF 3.1) is initially set to zero. PMF Data Output Mux Select 1 and 2 and Counter File or Status are active in Byte Count Translator (PMF 3.1). PMF Status Bits 0 through 4 and three zero-filled bits (Register 22 byte 0) are gated to Read Output Mux (PMF 3.5) through Status/Counter File Mux (PMF 3.5). Byte Counter (PMF 3.5) set to zero gates byte 0 through Read Output Mux as PMF Data Output Bits 0 through 7/8. PMF Data Output bits enter Channel Output Register via Data Muxes 2 and 3 (MAC 3.1).

If MAC Address is 21, Keypoint activates in PMF Interface Control (MAC 3.7) and enters Holding Register (PMF 3.0). Keypoint disables Read Output Mux Select 2 (PMF 3.5). After major cycle delay, Keypoint Delayed enables Keypoint Disassembly Mux output (PMF 3.5).

Disconnect Out enters Holding Register (PMF 3.0) as MAC Disconnect. Disconnect clears Keypoint Output Byte Counter (PMF 3.3).

If MAC Address is 21, Keypoint Delayed selects Keypoint Output Byte Count Bits 0 through 2 (initially equal to zero) in Byte Count Mux (PMF 3.1). Byte Select Bits 0 through 2 select Keypoint FIFO Byte 0 Bits 0 through 7 in Keypoint Disassembly Mux (PMF 3.5). Selected bits are from location 0 in Keypoint FIFO Buffer Memory (Register 21, PMF 3.3). Keypoint Buffer Bits 0 through 7 enter Parity Generator (PMF 3.5). PMF Read Data Bits 0 through 7/8 pass through Read Output Mux (PMF 3.5). PMF Data Output Bits 0 through 7/8 proceed to Channel Output Register (MAC 3.1) via Data Muxes 2 and 3 (MAC 3.1).

Second Activate Out generates Set Read Operation in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Set Read Operation activates Set MAC Read in Read, Write Control (MAC 3.6) translator.

2. Read Operation activates in Control FF (MAC 3.2).

MAC Read FF sets in Read, Write Control.

3. MAC Read activates MAC Read or Write in Data Ready Control (MAC 3.3).

4. MAC Read or Write partially enables Set Ready In To IOU in Data Ready Control.

MAC Read or Write also enters MAC Busy Control (MAC 3.7) and generates MAC Read, Write, or Clear, provided MAC Busy 1 and 2 are inactive. Refer to Read Element ID, Processor ID, Options Installed description, steps 2 through 7, for explanation of MAC Busy conflict-control network.

5. MAC Read, Write, or Clear activates Start MAC Timing Chain in Start MAC Timing Chain Translator (MAC 3.7).

6. MAC Timing Chain (MAC 3.7) starts.

If Register 21 read, refer to steps 7 through 21; if Register 22 read, refer to steps 22 through 34.

7. At MAC Timing Chain T4, PMF Transfer activates in PMF Interface Control (MAC 3.7).

Ready In Enable activates in Disconnect Translator (MAC 3.3).

8. Set Ready In To IOU activates in Data Ready Control (MAC 3.3).

9. Gate Channel Output Register loads PMF Data Output Bits 0 through 7/8 into Channel Output Register (MAC 3.1) by way of Data Muxes 2 and 3 (MAC 3.1).

10. PMF Transfer enters Holding Register in MAC Interface (PMF 3.0) and activates Transfer Byte in Transfer Byte Control (PMF 3.5).

11. At MAC Timing Chain T5, Clear MAC Busy 1 activates in Clear MAC Operation Translator (MAC 3.7) and Byte Transfer or Clear Error Complete activates in Read, Write Control translator (MAC 3.6).

Ready In To IOU from Data Ready Control (MAC 3.3) accompanies data byte to IOU.

12. Transfer Byte and Keypoint Delayed generate Keypoint Read in Keypoint Buffer Control (PMF 3.3).

13. MAC Read FF (MAC 3.6) clears.

MAC Busy 1 FF (MAC 3.7) clears, enabling CP to interrupt with MAC request.

14. Keypoint Read increments Keypoint Output Byte Counter (PMF 3.3).
15. MAC Busy 2 Register (MAC 3.7) clears.
16. IOU sends Ready Out to MAC. Refer to IOU Read, Write Operations, Ready Out description for details.
17. Combination of Read Operation and Ready 1 activates Set MAC Read in Read, Write Control (MAC 3.6).
18. Steps 2 through 17 repeat with variations required to select new byte until eight-byte transfer in complete. (Variation, step 2 - after Control FF sets (MAC 3.2) for first byte transfer, Read Operation remains active until new function enters MAC from IOU.) Keypoint Buffer Address Bits 0 through 3 (PMF 3.3) then increment to enable reading of next word. Cycle repeats until 16 words have been read from Register 21, or until an attempt is made to read the buffer when empty. In that case, PMF Disconnect enters the Disconnect Translator (MAC 3.3).
19. Disconnect Out enters Holding Register (PMF 3.0) as MAC Disconnect.
20. Disconnect clears Keypoint Output Byte Counter (PMF 3.3).
21. IOU terminates operation.
22. At MAC Timing Chain T4, PMF Transfer activates in PMF Interface Control (MAC 3.7).  
Ready In Enable activates in Disconnect Translator (MAC 3.3).
23. Set Ready In To IOU activates in Data Ready Control (MAC 3.3) translator.
24. Gate Channel Output Register loads PMF Data Output Bits 0 through 7/8 into Channel Output Register (MAC 3.1).
25. PMF Transfer enters Holding Register in MAC Interface (PMF 3.0) and activates Transfer Byte in Transfer Byte Control (PMF 3.5).
26. MAC Timing Chain T5 generates Clear MAC Busy 1 in Clear MAC Operation Translator (MAC 3.7) and activates Byte Transfer or Clear Error Complete in Read, Write Control translator (MAC 3.6).  
  
Ready In To IOU from Data Ready Control (MAC 3.3) accompanies data byte to IOU.
27. Byte Counter (PMF 3.5) increments (trailing edge of Transfer Byte).  
  
Transfer Byte departs Holding Register (PMF 3.5) and generates Reset or Increment in Initiate Control (PMF 3.1).
28. MAC Read FF (MAC 3.6) clears.  
  
MAC Busy 1 FF clears, enabling CP to interrupt with MAC request.
29. Initialization Byte Counter (PMF 3.1) increments.
30. MAC Busy 2 Register (MAC 3.7) clears.
31. IOU sends Ready Out to MAC. Refer to IOU Read, Write Operations, Ready Out description for details.

32. Combination of Read Operation and Ready 1 activates Set MAC Read in Read, Write Control (MAC 3.6).
33. Steps 2 through 6, 22 through 32 repeat with variation required to select new byte until 48-byte transfer is complete. (Variation, step 2 - after Control FF sets (MAC 3.2) for first byte transfer, Read Operation remains active until new function enters MAC from IOU.)

Order of byte selection is as follows:

- |                     |  |
|---------------------|--|
| Bytes 1 through 7   | From Counter File RAM (PMF 3.2) locations 1 through 7, selected in Status/Counter File Mux (PMF 3.5) and Read Output Mux (PMF 3.5). (Eight-bit bytes in locations 1 through 7 are stored in triplicate. Only one byte need be read from each 24-bit location.) |
| Bytes 8 through 15  | From A, B Counter Select Registers (PMF 3.5), selected in A, B Select Read Muxes (PMF 3.5) and Read Output Mux.  |
| Bytes 16 through 47 | <p>Bytes 16 through 19 are A0 Counter Output.</p> <p>Bytes 20 through 23 are B0 Counter Output.</p> <p>Bytes 24 through 27 are A1 Counter Output.</p> <p>•</p> <p>•</p> <p>Bytes 44 through 47 are B3 Counter Output.</p>                                      |

If the PMF is in operation, a read of Register 22 causes a single-byte transfer (status), after which PMF Disconnect proceeds to MAC.

Three high-order bytes of all eight counters are stored in Counter File (PMF 3.2) addresses 8 through 15 and selected in Status/Counter File Mux (PMF 3.5). Low-order byte of each counter (PMF 3.1) is selected in A, B Counter Disassembly Mux (PMF 3.5). All bytes proceed to MAC through Read Output Mux (PMF 3.5).

During last byte transfer, End Transfer FF sets in Transfer Byte Control (PMF 3.5). Subsequent PMF Transfer signal clears FF as last step in Register 22 read sequence.

34. IOU terminates operation.

#### PMF Register 22 Write (Function Code = 50, Address = 22)

The data path from the IOU to Register 22 is via Data Mux 6 (MAC 3.0). The IOU must write all 48 bytes of Register 22 to initiate PMF operations.

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations.

Opcode of five activates Operation Decode Equals 5 in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Type Code of five activates Gated Type Decode Equals 0.

Gated Type Decode Equals 0 disables Direct in Read and Write FFs translator (MAC 3.4).

Control Word 2 enters Address Incrementer (MAC 3.6). MAC Address Bits 0 through 7/8 enter MAC Address Decoder (MAC 3.5). Microcode Not Needed activates and partially enables Set MAC Write in Read, Write Control translator (MAC 3.6). MAC Address Equals 2X and MAC Address Equals X2 partially enable PMF Transfer in Performance Monitoring Facility Interface Control (MAC 3.7).

2. IOU sends Ready Out and data byte to MAC. Refer to IOU Read Write Operations, Ready Out description for details.

Data byte enters Maintenance Channel Data Parity Check (MAC 3.0) as Maintenance Channel Data Bits 0 through 7/8 and proceeds to Data Mux 6 (MAC 3.0).

3. PMF Data Input Bits 0 through 7/8 depart for Holding Register in MAC Interface (PMF 3.0).
4. From there, byte 0 enters second Holding Register (PMF 3.0). PMF Write Data Bits 0 through 7 proceed to Counter File RAM (PMF 3.2). Because Initialization Byte Counter (PMF 3.1) is preset to zero, Counter File RAM location 0 is addressed by output of Counter File Address Mux (PMF 3.2). Counter File Bytes 0 through 2 Select are active in Byte Count Translator (PMF 3.1), enabling byte 0 into all three byte positions in RAM location 0. Byte 0 is invalid. Subsequent data bytes all are valid.
5. Ready 1 activates Set Write Operation in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2).
6. Set Write Operation activates Set MAC Write in Read, Write Control translator (MAC 3.6).
7. MAC Write FF sets in Read, Write Control (MAC 3.6).
8. MAC Write activates MAC Read or Write in Data Ready Control (MAC 3.3).

MAC Write partially enables PMF Write in PMF Interface Control (MAC 3.7).

9. MAC Read or Write partially enables Set Ready In To IOU in Data Ready Control translator.

MAC Read or Write also enters MAC Busy Control (MAC 3.7) and generates MAC Read, Write, or Clear, provided MAC Busy 1 and 2 are inactive. Refer to Read Element ID, Processor ID, Options Installed description, steps 2 through 7, for explanation of MAC Busy conflict-control network.

10. MAC Read, Write, or Clear activates Start MAC Timing Chain in Start MAC Timing Chain Translator (MAC 3.7).
11. MAC Timing Chain (MAC 3.7) starts.
12. MAC Timing Chain T4 activates PMF Transfer and PMF Write in PMF Interface Control (MAC 3.7).

It also activates Ready In Enable in Disconnect Translator (MAC 3.3).

13. MAC Read or Write and Ready In Enable generate Set Ready In To IOU in Data Ready Control (MAC 3.3) translator.
14. PMF Transfer and PMF Write enter MAC Interface Holding Registers (PMF 3.0).
15. PMF Transfer activates Transfer Byte in Transfer Byte Control (PMF 3.5).

16. MAC Timing Chain T5 generates Byte Transfer or Clear Error Complete in Read, Write Control translator (MAC 3.6) and Clear MAC Busy 1 in Clear MAC Operation Translator (MAC 3.7).

Data Ready Control sends Ready In To IOU to IOU.

17. Write Delayed departs second Holding Register (PMF 3.0) and activates Write Counter File in Byte Count Translator (PMF 3.1).

Byte Counter (PMF 3.5) increments at trailing edge of Transfer Byte.

Transfer Byte departs Holding Register (PMF 3.5) and generates Reset or Increment in Initiate Control (PMF 3.1).

18. MAC Write FF (MAC 3.6) clears.
19. MAC Busy 1 FF (MAC 3.7) clears, enabling CP to interrupt with MAC Request.
20. PMF Write Data Bits 0 through 7 enter three bytes at location 0 of Counter File RAM (PMF 3.2).

Initialization Byte Counter Increments (PMF 3.1).

21. MAC Busy 2 Register (MAC 3.7) clears.
22. Steps 2 through 21 repeat with variations required to write new byte until 48-byte transfer is complete.

Order of byte entry is as follows:

Bytes 1 through 7      Enter Counter File RAM (PMF 3.2) locations 1 through 7. Single byte occupies all three byte positions at each location. Only one of the bytes is read at conclusion of a PMF operation. Bytes 1 through 3 also enter Register 22 (PMF 3.4). Bytes 4 through 7 are not used.

Bytes 8 through 15      Enter A, B Counter Select Registers (PMF 3.5).

Bytes 16 through 47      Bytes 16 through 19 enter A0 Counter.  
Bytes 20 through 23 enter B0 Counter.  
Bytes 24 through 27 enter A1 Counter.  
Bytes 28 through 31 enter B1 Counter.  
Bytes 32 through 35 enter A2 Counter.  
Bytes 36 through 39 enter B2 Counter.  
Bytes 40 through 43 enter A3 Counter.  
Bytes 44 through 47 enter B3 Counter.

Three high-order bytes of all eight counters enter Counter File RAM (PMF 3.2) addresses 8 through 15. Low-order byte enters counter, itself (PMF 3.1).

23. IOU terminates operation.

#### MICROCODE-ASSISTED READS, WRITES

The IOU requires CP microcode assistance to read or write various CP Type 0 registers not linked to MAC by the MAC Data Bus.

During microcode-assisted write operations, MAC initiates a micrand sequence to assemble data bytes from the IOU in a reserved Register File location. Once the word has been assembled, a second micrand sequence transfers the data to the desired register.

During microcode-assisted read operations, MAC initiates a micrand sequence which sends the contents of the desired register to the CST Disassembly Network (CST 3.5). MAC then controls disassembly and transfer of the data from the CST Disassembly Network to the IOU via the Maintenance Channel. Table 2-1 indicates the registers which MAC requires CP microcode assistance to read or write and lists access restrictions.

#### Microcode-Assisted Read (Function Code = 40, Address = 13, 4X-6X, CX or EX)

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations.

Opcode of four activates Operation Decode Equals 4 in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Type Code of zero activates Gated Type Decode Equals 0.

Gated Type Decode Equals 0 disables Direct in Read and Write FFs translator (MAC 3.4).

Control Word 2 enters Address Incrementer (MAC 3.6). MAC Address Bits 0 through 7/8 enter MAC Address Decoder (MAC 3.5). Provided CP is not already engaged in another microcode-assisted operation, Microcode Not Needed remains inactive. Microcode Not Needed partially enables Set Microcode-Assisted Read in Read, Write Control translator (MAC 3.6).

Mux 5 Select 1 is active in Data Mux 4, 5 Select Translator (MAC 3.0), gating MAC Data Bus Bits 0 through 7/8 from CST Disassembly Network (CST 3.5) into Data Mux 5 (MAC 3.0).

Second Activate Out generates Set Read Operation in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Set Read Operation, inactive Direct and inactive Microcode Not Needed activate Set Microcode-Assisted Read in Read, Write Control translator (MAC 3.6).

2. Microcode Read FF sets (MAC 3.6).
3. Microcode Read activates Mux Select 2 and MAC Request in Microcode Request Control (MAC 3.6).
4. MAC Microcode Address Bits 0 through 7/8 (equal to  $30_{16}$ ) proceed to MAC Left Most Byte Mux Register (CST 3.3) and from there to Micrand Address Mux 2 (CST 3.4) as MAC Microcode Address Bits 3 through 10/Parity. MAC Microcode Address Bits 0 through 2 are zero-filled.  
  
MAC Request enters MAC Request Control (CST 3.0) to produce MAC Request A, B, or C when current CP instruction exits.
5. When current CP instruction or sequence exits, MAC Request A, B, or C activates.
6. MAC Request and Exit or Halted proceeds to Holding Register in MAC Request Control (CST 3.0).

MAC Request A, B, or C activates Select 2 in Micrand Address Mux 2 Select (CST 3.4).



7. MAC Microcode Address Bits 0 through 10/Parity proceed through Micrand Address Mux 2 and 1 to Micrand Address Mux Register input (CST 3.4).
8. Enable Micrand Address Mux Register loads CST RAM address into Micrand Address Mux Register. CST Address Bits 0 through 10 address CST RAM (CST 3.5) location  $30_{16}$ .
9. MAC Request and Exit or Halted enters MAC Request Control Holding Register (CST 3.0).
10. MAC In Pipe activates and proceeds to OR gate (MAC 3.6).
11. MAC In Pipe or Off Line gates MAC Address Bits 0 through 7 from Reference ROM Address Mux (MAC 3.6) to Reference ROM (MAC 3.6) as Reference ROM Address Bits 0 through 7.
12. Reference ROM Bits 0 through 5/6 proceed to Address Mux 2 (CST 3.4).
13. Micrand sequence starting at address  $30_{16}$  jumps to CST RAM address  $200_{16}$  plus Reference ROM output.
14. Micrands then transfer desired register's contents to Disassembly Register in CST Disassembly Network (CST 3.5) via Register File (OPI 3.5) location 9 and Read Data Register and Parity Check (OPI 3.5).
15. Set MAC Busy activates in LRT and MIS Field Decoders (CST 3.1).
16. Set MAC Busy sets MAC Busy 1 FF (MAC 3.7) to reserve disassembly network and MAC Data Bus.
17. CP then sends MAC Opcode Bits 0 through 2 and MAC Response/Request to Instruction Unit/MAC Operation Code Decoder (MAC 3.7).
18. MAC Opcode bits enter MAC Opcode Register.  
  
Response/Request FF sets (MAC 3.7) and MAC Response/Request 1 exits Holding Register (MAC 3.7).
19. MAC Operation Equals 7 and MAC Operation Equals 2 or 7 activate.  
  
MAC Operation Bit 0 disables Select PFS Bus, which enables CST Byte Count Bit 0 in CST Disassembly Control (MAC 3.8).
20. MAC Operation Equals 7, MAC Request, and MAC Response/Request 1 generate Start MAC Timing Chain in Start MAC Timing Chain Translator (MAC 3.7).  
  
MAC Operation Equals 7 partially enables Set Ready In To IOU in Data Ready Control translator (MAC 3.3).  
  
MAC Operation Equals 2 or 7 activates Microcode Operation Complete in Read, Write Control translator (MAC 3.6).
21. MAC Timing Chain starts.  
  
Microcode Read FF (MAC 3.6) clears.
22. MAC Request deactivates in Microcode Request Control (MAC 3.6).
23. At MAC Timing Chain T4, Ready In Enable activates in Disconnect Translator (MAC 3.3).

24. Ready In Enable and MAC Operation Equals 7 generate Set Ready In To IOU in Data Ready Control translator (MAC 3.3).
25. Gate Channel Output Register loads byte from CST Disassembly Network (CST 3.5) into Channel Output Register (MAC 3.1).

Because MAC Byte Counter (MAC 3.8) is initially set to zero, CST Byte Count of zero selects byte 0 in Disassembly Register (CST 3.5) from time register is loaded until MAC Byte Counter increments.

Path from CST Disassembly Network to Channel Output Register is via MAC Data Bus, Data Mux 5 (MAC 3.0) and Data Mux 3 (MAC 3.1).

26. MAC Timing Chain T5 activates and increments MAC Byte Counter (MAC 3.8) to select next byte in CST Disassembly Network.

Ready In To IOU from Data Ready Control (MAC 3.3) accompanies data byte to IOU.

27. IOU sends Ready Out to MAC. Refer to IOU Read, Write Operations, Ready Out description for details.
28. Ready 1 activates Start MAC Timing Chain in Start MAC Timing Chain Translator (MAC 3.7).
29. MAC Timing Chain starts.
30. Steps 23 through 29 repeat until all eight bytes are transferred from CST Disassembly Network to IOU.
31. At MAC Timing Chain T5 during last byte transfer (step 26 when MAC Byte Counter equals seven), Clear MAC Busy 1 and Clear MAC Operation activates in Clear MAC Operation Translator (MAC 3.7).
32. Clear MAC Operation 1 resets MAC Opcode Register (MAC 3.7).  
  
Clear MAC Operation resets Response/Request FF (MAC 3.7) and Clear MAC Busy 1 resets MAC Busy 1 FF (MAC 3.7).
33. MAC Busy 2 Register (MAC 3.7) clears.
34. IOU terminates operation.

#### Microcode-Assisted Write (Function Code = 50, Address = 13, 4X-6X, CX or EX)

During a write access, MAC initiates a CP micrand sequence to assemble bytes from the IOU in a 64-bit location in the Register File (OPI 3.5). Once the 64-bit word has been assembled, another micrand sequence transfers the data to the desired register.

1. Function Code and Control Words 1 and 2 enter MAC. Refer to Read, Write Initial Steps for details about events common to all IOU operations.

Opcode of five activates Operation Decode Equals 5 in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2). Type Code of zero activates Gated Type Decode Equals 0.

Operation Decode Equals 5 partially enables Set Write Operation in Maintenance Channel Opcode, Type Code Register, Decoder.

Gated Type Decode Equals 0 disables Direct in Read and Write FFs translator (MAC 3.4).

Control Word 2 enters Address Incrementer (MAC 3.6). MAC Address Bits 0 through 7/8 enter MAC Address Decoder (MAC 3.5). Provided CP is not already engaged in another microcode-assisted operation, Microcode Not Needed remains inactive. Microcode Not Needed partially enables Set Microcode-Assisted Byte Store and Microcode-Assisted Write in Read, Write Control (MAC 3.6).

2. IOU sends Ready Out and data byte to MAC. Refer to IOU Read, Write Operations, Ready Out description for details.

Data byte enters Maintenance Channel Parity Check (MAC 3.0) as Maintenance Channel Data Bits 0 through 7/8 and proceeds to Data Mux 6 (MAC 3.0). Bit 8 of even byte enters MAC Even Byte Parity Register.

3. MAC Data Bus Bits 0 through 7/8, 9 go to Register Write Data Mux (OPI 3.5) from Data Mux 6.
4. Ready 1 activates Set Write Operation in Maintenance Channel Opcode, Type Code Register, Decoder (MAC 3.2).
5. Set Write Operation activates Set Microcode-Assisted Byte Store in Read, Write Control translator (MAC 3.6).
6. Microcode Byte Store FF sets (MAC 3.6).
7. Microcode Byte Store activates Mux Select 1 and MAC Request in Microcode Request Control (MAC 3.6).

Microcode Byte Store also activates Decoder Enable in Register File Byte Selector (MAC 3.8).

(Register File Byte 0 and 1 Select activate in Register File Byte Select Decoder (MAC 3.8) because MAC Byte Counter is initially set to zero. During second byte transfer, only Register File Byte 1 Select activates. For discussion of Register File Byte N Select control of bytes 2 through 7 entry into Register File, refer to Register File Write, step 21.)

8. MAC Request enters MAC Request Control (CST 3.0) to produce MAC Request A, B, or C when current CP instruction or sequence exits.

MAC Microcode Address Bits 0 through 7/8 (equal to  $32_{16}$  because active Mux Select 1 enters Microcode Request Control (MAC 3.6)) proceed from Microcode Request Control to MAC Left Most Byte Mux Register (CST 3.3).

9. From there they enter to Micrand Address Mux 2 (CST 3.4) as MAC Microcode Address Bits 3 through 10/Parity. MAC Microcode Address Bits 0 through 2 are zero-filled.
10. When current instruction exits, CST Address Bits 0 through 10 address CST RAM (CST 3.5) location  $32_{16}$ . Refer to Microcode-Assisted Read, steps 5 through 8 for details.
11. Micrand sequence starting at address  $32_{16}$  stores byte on MAC Data Bus at Register File location 1 (OPI 3.5).

Even Byte locations in RAM provide no space for parity. Parity for odd byte and saved parity for even byte are stored in RAM with odd byte (refer to Soft Control Memory Write, step 16, for further explanation).

12. MAC Byte Store Response returns from CST Disassembly Network Interface (OPI 3.4) to Byte Store Timing Delay (MAC 3.6). CP resumes interrupted instruction sequence.
13. Byte Store Response enters Holding Register (MAC 3.6) and activates Microcode Operation Complete in Read, Write Control translator (MAC 3.6).  
  
Byte Store Response also activates Set Ready In To IOU in Data Ready Control translator (MAC 3.3).
14. Microcode Byte Store FF clears.  
  
Data Ready Control sends Ready In To IOU to IOU.  
  
Delayed Byte Store Response increments MAC Byte Counter (MAC 3.8), resulting in new Register File Byte Select decode (MAC 3.8).
15. Steps 2 through 14 repeat until eight bytes have entered Register File location 1.
16. During eighth byte-store cycle, returning Byte Store Response (step 13) fails to activate Set Ready In To IOU in Data Ready Control (MAC 3.3).  
  
Byte Store Response activates Set Microcode-Assisted Write in Read, Write Control translator (MAC 3.6).
17. Microcode Write FF sets (MAC 3.6).  
  
Microcode Byte Store FF clears.
18. Microcode Write activates Mux Select 3 and MAC Request in Microcode Request Control translator (MAC 3.6).
19. MAC Microcode Address Bits 0 through 7/8 (equal to  $31_{16}$ ) proceed to Micrand Address Mux 2 (CST 3.4). Refer to steps 8, 9 for details.  
  
MAC Request enters MAC Request Control (CST 3.0) to produce MAC Request A, B, or C when current CP instruction exits.
20. When current CP instruction or sequence exits, CST Address Bits 0 through 10 address CST RAM (CST 3.5) location  $31_{16}$ . Refer to Microcode-Assisted Read, steps 5 through 8, for details.
21. MAC Address Bits 0 through 7 enter Reference ROM (MAC 3.6). Refer to Microcode-Assisted Read, steps 9 through 11, for details.
22. Reference ROM Bits 0 through 5/6 proceed to Address Mux 2 (CST 3.4).
23. Micrand sequence starting at address  $31_{16}$  transfers word assembled at Register File location 1 to Register File location 8.
24. Sequence jumps to CST RAM Address  $3C0_{16}$  plus index provided by Reference ROM Bits 0 through 5. Addressed Micrand sequence transfers data from Register File location 8 into desired register.  
  
Micrand Sequence generates MAC Response/Request in MAC Opcode/Response Generator (OPI 3.15) and sends it with MAC Opcode Bits 0 through 2 to Instruction Unit/MAC Operation Code Decoder (MAC 3.7). CP exits to interrupted instruction sequence.
25. MAC Opcode Bits 0 through 2 enter MAC Opcode Register (MAC 3.7).  
  
MAC Response/Request sets Response/Request FF (MAC 3.7).

26. MAC Operation Equals 2 and MAC Operation Equals 2 or 7 activate.
27. MAC Operation Equals 2 generates Clear MAC Busy 1 and Clear MAC Operation in Clear MAC Operation Translator (MAC 3.7).  
  
It also activates Set Ready In To IOU in Data Ready Control translator (MAC 3.3).  
  
MAC Operation Equals 2 or 7 activates Microcode Operation Complete in Read, Write Control translator (MAC 3.6).
28. Clear MAC Operation 1 resets MAC Opcode Register (MAC 3.7).  
  
Clear MAC Operation resets Response/Request FF (MAC 3.7).  
  
Data Ready Control sends Ready In To IOU to IOU.  
  
Microcode Write FF clears.
29. IOU terminates operation.

#### CP READ, WRITE OPERATIONS

The CP uses C180 Copy State (OE, OF) instructions to read or write Type 0 registers (either MAC or CP-resident). The data transfer is between the Xk Register in the Register File and the register designated by the contents of the Xj Register.

In all cases, Copy State instructions use the Reference ROM (MAC 3.6). An index from the Reference ROM, addressed by OPI Copy Address, is added to a constant. The result is the CST RAM (CST 3.5) address of a read or write micrand sequence.

In addition to addressing the Reference ROM, certain Copy State instructions use the resulting micrand sequence to read or write registers residing in MAC - the Element ID, Processor ID, Options Installed, DEC, PTM, or PFS registers. Because the IOU also may read or write these registers, the CP must test MAC conflict-control circuitry before proceeding. Refer to Read Element ID, Processor ID, Options Installed description, steps 3 through 7, for explanation.

Execution of a C180 Keypoint (B1) instruction may require the CP to write Keypoint Class and Keypoint Code data into PMF Register 21. This instruction likewise uses MAC data paths and conflict-control circuitry.

#### MAC-RESIDENT REGISTER READ

The CP supplies MAC with the OPI Copy Address (Xj). MAC selects and disassembles eight bytes of data from the addressed register. The data bytes proceed to OPI via the MAC Data Bus and are reassembled in the Register File (OPI 3.5) for entry into the Register Xk.

1. Initial Micrands of Copy from State (OE) instruction load contents of Xj Register into Reference ROM Address Register (OPI 3.16). OPI Copy Address Bits 0 through 7/8 proceed to MAC Address Decoder input (MAC 3.5) and to Reference ROM (MAC 3.6) through Reference ROM Address Mux (MAC 3.6).
2. Micrand sequence branches to CST RAM (CST 3.5) address  $200_{16}$  plus address index from Reference ROM.

3. Micrand sequence which follows tests output of MAC Busy 1 FF at Condition Sensing Mux (CST 3.2). If MAC is not busy, Set MAC Busy activates in LRT Field Decoder (CST 3.1).
4. Set MAC Busy sets MAC Busy 1 FF (Mac 3.7) and blocks IOU from reading or writing MAC-resident registers for as long as FF set.
5. MAC Busy 2 Register (MAC 3.7) sets.
6. MAC Opcode Bits 0 through 2 enter MAC Opcode Register (MAC 3.7) and MAC Response/Request enters Response/Request FF (MAC 3.7) and Holding Register (MAC 3.7).
7. CP Micrand sequence loops until MAC Busy 1 FF clears at end of MAC-controlled read operation.
8. MAC Operation Equals 1 activates Decoder Enable in Register File Byte Selector (MAC 3.8).

Inactive MAC Operation Bit 0 activates Select PFS Bus in CST Disassembly Control (MAC 3.8).

MAC Operation Equals 1 or 6 partially enables Clear MAC Busy 1 in Clear MAC Operation Translator (MAC 3.7) and activates Mode Select Enables in PFS Register Select Translator (MAC 3.5).

MAC Operation Equals 1 or 6 and MAC Response/Request 1 activate Start MAC Timing Chain in Start MAC Timing Chain Translator (MAC 3.7).

MAC Response/Request from Pipeline activates IU Reserve.

9. IU Reserve enables output of Data Mux 5 to enter Data Mux 6 (MAC 3.0) for transfer to Register File (OPI 3.5) once read begins.

IU Reserve gates OPI Copy Address Bits 0 through 7/8 to MAC Address Decoder (MAC 3.5). Address of MAC-resident register to be read is translated. Address decode determines which MAC-resident data is to be selected in Data Mux 5. Additionally, address determines if DEC, PTM, or PFS data is to be read, routed through CST Disassembly Network (CST 3.5), and returned to Data Mux 5 as MAC Data Bus Bits 0 through 7/8.

IU Reserve enables Increment Counter in Instruction Unit Byte Counter translator (MAC 3.8) and gates counter output (initially set to zero) to Register File Byte Select Decoder (MAC 3.8) to control selection of bytes for input to Register File.

MAC/IU Byte Control Bits 0 through 2 also control order of byte selection from register being read, according to pattern of MAC Reads. Refer to MAC Reads, Writes - Read Element ID, Processor ID, Option Installed; Read DEC, PTM, PFS Register for details of byte selection.

10. MAC Timing Chain (MAC 3.7) starts.
11. If DEC, PTM, or PFS register is being read, MAC Timing Chain T1 (followed by MAC Timing Chain T2 through T4) enables board selection. Refer to Read DEC, PTM, PFS Register, steps 6 and 28, for details.
12. During MAC Timing Chain T2, selected DEC, PTM, or PFS byte passes through CST Disassembly Network (CST 3.5) as MAC Data Bus Bits 0 through 7/8. Data byte enters Data Mux 5 (MAC 3.0). Refer to Read DEC, PTM, PFS Register, steps 12 through 17 and 30 through 34, for details.

First Element ID, Processor ID or Options Installed data byte is selected in Data Mux 5 from time address is translated (step 9). Subsequent bytes become available when Instruction Unit Byte Counter increments (step 20).

13. Data Mux 5 Bits 0 through 7/8 enter Data Mux 6 (MAC 3.0).

Because parity for odd and even bytes is stored with odd byte in Register File, even byte's bit 8 enters Instruction Even Byte Parity Register (MAC 3.0). Refer to Soft Control Memory Write, step 16, for details.

14. MAC Data Bus Bits 0 through 7/8, 9 depart Data Mux 6 for Register File Write Data Mux (OPI 3.5).
15. Select MAC Data, activated by General Micrand field (OPI 3.4), gates data byte to Register File RAM (OPI 3.5).
16. At MAC Timing Chain T3, Register File Go activates in Register File Go Control translator (MAC 3.8).
17. Register File Go enters Holding Register (MAC 3.8) and proceeds to Register File Write Control (OPI 3.4).
18. Write Register File activates and gates Register File Bytes 0 through 7 Selects to Register File location 0 through Holding Register (OPI 3.4).
19. Whichever Register File Bytes 0 through 7 Select signals are active gate MAC Data Bus Bits 0 through 7/8, 9 into appropriate byte positions in Register File RAM location 0. For explanation of byte entry order into Register File RAM, refer to Register File Write, step 21, description.
20. At MAC Timing Chain T5, Start MAC Timing Chain activates in Start MAC Timing Chain Translator (MAC 3.7).

MAC Timing Chain T5 increments Instruction Unit Byte Counter (MAC 3.8) to select next byte to be transferred and next Register File RAM byte position to be entered.

21. Steps 10 through 20 repeat until eight-byte transfer is complete.
22. During step 20 of final byte transfer, MAC IU/Byte Count Equals 7 disables regeneration of Start MAC Timing Chain and activates Clear MAC Busy in Clear MAC Operation Translator (MAC 3.7).
23. MAC Busy 1 FF (MAC 3.7) clears.

MAC Opcode Register and Response/Request FF (MAC 3.7) also clear.

24. Micrand sequence senses clear state of MAC Busy 1 FF and proceeds to transfer assembled data word from Register File location 0 to Xk Register specified by instruction.
25. Instruction then exits.

#### MAC-RESIDENT REGISTER WRITE (PFS, PTM ONLY)

The CP transfers write data from the Xk Register in the Register File (OPI 3.5) to the CST Disassembly Network (CST 3.5). MAC controls disassembly of the data word into eight bytes for transfer on the MAC Data Bus. The disassembled bytes enter the MAC-resident register addressed by the CP.

1. Initial micrands of Copy to State (OF) instruction load contents of Xj Register into Reference ROM Address Register (OPI 3.16).
2. OPI Copy Address Bits 0 through 7/8 then proceed to MAC Address Decoder input (MAC 3.5) and Reference ROM (MAC 3.6) through Reference ROM Address Mux (MAC 3.6).
3. Micrand sequence branches to CST RAM (CST 3.5) address  $3C0_{16}$  plus address index from Reference ROM.
4. Micrand sequence which follows tests output of MAC Busy 1 FF at Condition Sensing Mux (CST 3.2). If MAC is not busy, Set MAC Busy activates in LRT Field Decoder (CST 3.1).
5. Set MAC Busy sets MAC Busy 1 FF (MAC 3.7) and blocks IOU from reading or writing MAC-resident registers for as long as FF is set.
6. MAC Busy 2 Register (MAC 3.7) sets.
7. Contents of Xk Register specified by instruction enter Disassembly Register in CST Disassembly Network (CST 3.5).
8. MAC Opcode Bits 0 through 2 enter MAC Opcode Register (MAC 3.7), and MAC Response/Request enters Response/Request FF (MAC 3.7) and Holding Register (MAC 3.7).
9. CP Micrand sequence loops until MAC Busy 1 FF clears at end of MAC-controlled write operation.
10. MAC Operation Equals 6 activates Write Mode in PFS Register Select Translator (MAC 3.5).

MAC Operation Bit 0 disables Select PFS Bus in CST Disassembly Control (MAC 3.8), which enables CST Byte Count Bit 0.

MAC Operation Equals 1 or 6 partially enables Clear MAC Busy 1 in Clear MAC Operation Translator (MAC 3.7) and activates Mode Select Enable in PFS Register Select Translator.

MAC Operation Equals 1 or 6 and MAC Response/Request 1 activate Start MAC Timing Chain in Start MAC Timing Chain Translator (MAC 3.7).

MAC Response/Request from Pipeline activates IU Reserve.

11. IU Reserve enables output of Data Mux 5 to enter Data Mux 6 (MAC 3.0). Data is transferred from CST Disassembly Network (CST 3.5) to PFS or PTM register along this path once write begins.

IU Reserve gates OPI Copy Address Bits 0 through 7/8 to MAC Address Decoder (MAC 3.5). Address of MAC-resident register to be written is translated.

IU Reserve enables Increment Counter in Instruction Unit Byte Counter translator (MAC 3.8) and gates counter output through Register File Byte Selector as MAC/IU Byte Count Bits 0 through 2.

12. MAC/IU Byte Count Bits 0 through 2 (initially set to zero) go to CST Disassembly Control (MAC 3.8). CST Byte Count Bits 0 through 2 enter CST Disassembly Network (CST 3.5) to control byte selection during write operation.

OPI Copy Address decode activates Mux 5 Select 1 in Data Mux 4, 5 Select Translator (MAC 3.0). Data Mux 5 (MAC 3.0) selects MAC Data Bus Bits 0 through 7/8.



13. MAC Timing Chain (MAC 3.7) starts.
14. If PFS register is being written, MAC Timing Chain T1 (followed by MAC timing Chain T2-T4) gates Address Bits 4 through 6 to PFS Board Select Decoder (MAC 3.5). Address decode determines which PFS register receives write data. Refer to MAC Reads, Writes, Write DEC, PTM, PFS Register, steps 27 through 30, for details.  
  
If PTM register is being written, MAC Timing Chain T1 (followed by MAC Timing Chain T2-T4) gate MAC/IU Byte Count Bits 0 and 1 to PTM Board Select Decoder (MAC 3.5). MAC/IU Byte Count Bits 0 through 2 (initially equal to zero) control order of byte entry into PTM register, according to pattern of MAC writes. Refer to MAC Read, Writes, Write DEC, PTM, PFS Registers, steps 11 through 25, for details.
15. If PFS register is being written, MAC Data Bus Bit 7 sets or clears selected PFS Register after entering Set/Clear PFS Control translator (MAC 3.10) as Board N PFS Bus Data Input Bit 7.  
  
If PTM register is being written, MAC Data Bus Bits 0 through 7/9 from Data Mux 6 (MAC 3.0) enter PTM Register (MAC 3.11) as PFS Bus Data Bits 0 through 7/8.
16. MAC Timing Chain T5 activates Start MAC Timing Chain in Start MAC Timing Chain Translator (MAC 3.7) and increments Instruction Unit Byte Counter (MAC 3.8).
17. Incremented CST Byte Count Bits 0 through 2 select next byte for output to MAC-resident register from CST Disassembly Network (CST 3.5).
18. Steps 13 through 17 repeat until eight byte transfer is complete.  
  
Condition of last byte's low-order bit determines final state of selected even PFS register. If bit is logical one, all bits in register are set. If bit is logical zero, all bits are cleared.  
  
Condition of second byte's low-order bit determines final state of selected odd PFS register. PFS Odd Zero Fill disables Write Mode in PFS Register Select Translator (MAC 3.5) after second byte.
19. During step 16 of final byte transfer, MAC IU/Byte Count Equals 7 disables regeneration of Start MAC Timing Chain and activates Clear MAC Busy 1 in Clear MAC Operation Translator (MAC 3.7).
20. MAC Busy 1 FF clears (MAC 3.7).  
  
MAC Opcode Register and Response/Request FF (MAC 3.7) also clear.
21. Micrand sequence senses clear state of MAC Busy 1 FF. Instruction exits.

#### PMF REGISTER 21 WRITE

1. Initial Micrands of Keypoint (B1) instruction add constant Q from instruction and contents of Xk Register to produce 32-bit Keypoint Code. Instruction's four-bit j field is Keypoint Class.
2. If Keypoint Request Flag (DAI Bit 63) is set, Keypoint Code and Keypoint Class enter Register File location OF.
3. Micrand sequence that follows tests output of MAC Busy 1 FF at Condition Sensing Mux (CST 3.2). If MAC is not busy, Set MAC Busy activates in LRT Field Decoder (CST 3.1).

4. Set MAC Busy sets MAC Busy 1 FF (MAC 3.7) and blocks IOU from reading or writing MAC-resident registers for as long as FF is set.
5. MAC Busy 2 Register (MAC 3.7) sets.
6. Contents of Register File location OF enter Disassembly Register in CST Disassembly Network (CST 3.5). Keypoint Code occupies low-order 32 bit positions. Keypoint Class occupies four adjacent high-order bit positions. High-order 28 bits are sent to PMF, but not written into Register 21.
7. MAC Opcode Bits 0 through 2 enter MAC Opcode Register (MAC 3.7) and MAC Response/Request enters Response/Request enters Response/Request FF (MAC 3.7) and Holding Register (MAC 3.7).
8. MAC Operation Equals 4 activates Keypoint and PMF Write in PMF Interface Control (MAC 3.7).

MAC Operation Equals 4 also partially enables Clear MAC Busy 1 in Clear MAC Operation Translator (MAC 3.7) and Increment Counter in Instruction Unit Byte Counter translator (MAC 3.8).

MAC Operation Bit 0 disables Select PFS Bus in CST Disassembly Control (MAC 3.8), which enables CST Byte Count Bit 0.

MAC Response/Request from Pipeline activates IU Reserve (MAC 3.7).

9. IU Reserve enables output of Data Mux 5 to enter Data Mux 6 (MAC 3.0) for data transfer from CST Disassembly Network (CST 3.5) to Keypoint Buffer (PMF 3.3).

IU Reserve also enables Instruction Unit Byte Count Bits 0 through 2 to become MAC/IU Byte Count Bits 0 through 2 in Register File Byte Selector (MAC 3.8). On entering CST Disassembly Control (MAC 3.8), MAC/IU Byte Count Bits 0 through 2 become CST Byte Count Bits 0 through 2, which control byte selection in CST Disassembly Network (CST 3.5). CST Byte Count equals zero initially.

10. Keypoint and PMF Write enter Holding Registers in MAC Interface (PMF 3.0).

PMF Data Input Bits 0 through 7/8 enter Holding Register (PMF 3.0).

11. Instruction Unit Byte Counter (MAC 3.8) increments to select next byte for input from CST Disassembly Network (CST 3.5).

12. PMF Write and Keypoint enter second set of Holding Registers (PMF 3.0).

PMF Data Input Bits 0 through 7 enter second Holding Register (PMF 3.0) and become PMF Write Data Bits 0 through 7, which are not used.

PMF Data Input Bits 0 through 7/8 (second invalid byte from CST Disassembly Network) enter first Holding Register (PMF 3.0).

13. Write Delayed and Keypoint Delayed generate Write Register 21 (PMF 3.1).

Write Delayed and Keypoint Delayed generate Write Keypoint (PMF 3.3), which gates Byte Select Bits 0 through 2 from Byte Count Mux (PMF 3.1) into Decoder (PMF 3.3).

14. Keypoint Delayed and Write Register 21 gate Keypoint Input Byte Count Bits 0 through 2 into Byte Count Mux (PMF 3.1).

Write Keypoint and PMF Write partially enable increment signal to Keypoint Input Byte Counter (PMF 3.3), source of Register 21 write byte count.

15. Because Keypoint Input Byte Count is zero initially, Load Keypoint Byte 0 activates in Decoder (PMF 3.3). Load Keypoint Byte 0 gates contents of 1 us Second Timer (PMF 3.3) into Keypoint FIFO Buffer Memory (PMF 3.3) location addressed by Keypoint Write Address Counter (PMF 3.3). Timer output enters three and one-half high-order byte positions.
16. Step 11 repeats.
17. Keypoint Input Byte Counter (PMF 3.3) increments.
18. Second invalid data byte enters second Holding Register (PMF 3.0) and becomes PMF Write Data Bits 0 through 7.  
  
Third invalid data byte enters first Holding Register (PMF 3.0).
19. Steps 16 through 18 repeat. Third invalid data byte enters second Holding Register. Fourth data byte enters first Holding Register.
20. Step 11 repeats.
21. Keypoint Input Byte Counter (PMF 3.3) increments.
22. Keypoint Input Byte Count Bits 0 through 2 enter Decoder (PMF 3.3) per steps 13 through 15. Load Keypoint Byte 3 activates.
23. Fourth data byte (keypoint class in four low-order bit positions) enters second Holding Register (PMF 3.0).  
  
Fifth data byte enters first Holding Register (PMF 3.0).
24. Byte 3 Bits 4 through 7 enter addressed location in Keypoint FIFO Buffer Memory (PMF 3.3).
25. Steps 20 through 24 repeat with variations required to enter next four bytes (keypoint code) into addressed memory location.  
  
Clear MAC Busy 1 activates in Clear MAC Operation Translator (MAC 3.7) during step 20 of seventh byte transfer into memory location. (Disassembly count leads memory load count by one.) During step 20 of eighth byte transfer, MAC Busy 1 FF (MAC 3.7) clears, as do MAC Opcode Register and Response/Response FF (MAC 3.7).



**SECTION 3**

**INSTRUCTION ISSUE**



---

This section describes Instruction Fetch (IF), Control Store (CST), Instruction Control Pipeline (ICP), and Operand Issue (OPI). IF reads instruction words from memory and disassembles them into individual instructions. CST supplies the microcode to control instruction execution. ICP provides registers to hold instructions and associated micrands while the instructions are in different stages of their execution. OPI contains the Register File and provides control for reading operands and storing results.





PART 1

INSTRUCTION FETCH (IF)



The following text is to be used in conjunction with the IF 1.0 diagram for an explanation of IF operations.

After IF reads instruction words from LM, IF assembles individual instructions and stores them in five buffer ranks. The instructions pass through IF Instruction Buffer Ranks 2, 3, 10, 11, and 12, proceeding from Instruction Buffer Rank 12 to CST and ICP to begin execution. IF can fill its buffer ranks at the rate of two per major cycle, twice the maximum instruction execution rate in the IU.

In addition to addressing successive instruction words during a given instruction sequence (RNI) with P bits, the Branch Address Adder calculates and issues the addresses of new sequences specified by branch instructions.

#### BRANCH INSTRUCTIONS

Four major characteristics establish two categories of branch instructions in C170 mode and C180 mode.

First, branch instructions are either conditional or unconditional. Conditional branch instructions depend on the presence of a stated requirement (usually a specified relationship between two operands). IF presumes initially the branch condition is met and addresses the new instruction sequence. If ALN (which tests the branch conditions) later determines the condition has not been met, IF addresses the instruction following the branch instruction in the original sequence. IF also clears the instruction pipeline (in IF and ICP) of any instructions from the sequence it branched to. This operation is referred to as unbranching.

Unconditional branch instructions depend on no external requirements. The CP automatically proceeds with the new instruction sequence.

Secondly, branch instructions are either relative or indexed. IF calculates a relative branch address, while ALN calculates an indexed branch address.

In C170 mode, the Branch Address Adder calculates the relative branch address from eight times RAC (a constant from the exchange package) and eight times K (a constant field from the instruction). In C180 mode, the adder produces a relative branch address from P (the address of the branch instruction) and two times Q (a constant field from the branch instruction). In C170 mode the addresses are defined as relative to RAC; in C180 mode the addresses are defined as relative to P.

By contrast, indexed branch addresses are calculated from the contents of one or more Register File locations. For example, ALN computes the indexed branch address for a C170 02 instruction from the contents of the Bj Register and K. The C180 2E and 2F instructions also are indexed branch instructions.

The Branch Address Adder calculates a relative branch address at instruction execution Time 3. ALN calculates and sends the indexed branch address to IF at instruction execution Time 50.

The following is a list of C170 and C180 mode branch instructions. Conditional branch addresses are always relative and unconditional branch addresses are indexed.

## C170 MODE

### CONDITIONAL, RELATIVE

030 Branch to K if  $(X_j) = 0$   
031 Branch to K if  $(X_j) \neq 0$   
032 Branch to K if  $(X_j)$  positive  
033 Branch to K if  $(X_j)$  negative  
034 Branch to K if  $(X_j)$  in range  
035 Branch to K if  $(X_j)$  out of range  
036 Branch to K if  $(X_j)$  definite  
037 Branch to K if  $(X_j)$  indefinite  
04 Branch to K if  $(B_i) = (B_j)$   
05 Branch to K if  $(B_i) \neq (B_j)$   
06 Branch to K if  $(B_i) \geq (B_j)$   
07 Branch to K if  $(B_i) < (B_j)$

### UNCONDITIONAL, INDEXED

02 Jump to  $(B_i) + K$

## C180 MODE

### CONDITIONAL, RELATIVE

90 Branch to P displaced by 2Q if  $(X_j)$  Right =  $(X_k)$  Right  
91 Branch to P displaced by 2Q if  $(X_j)$  Right  $\neq$   $(X_k)$  Right  
92 Branch to P displaced by 2Q if  $(X_j)$  Right  $>$   $(X_k)$  Right  
93 Branch to P displaced by 2Q if  $(X_j)$  Right  $\geq$   $(X_k)$  Right  
94 Branch to P displaced by 2Q if  $(X_j) = (X_k)$   
95 Branch to P displaced by 2Q if  $(X_j) \neq (X_k)$   
96 Branch to P displaced by 2Q if  $(X_j) > (X_k)$   
97 Branch to P displaced by 2Q if  $(X_j) \geq (X_k)$   
98 Branch to P displaced by 2Q if Floating-Point  $(X_j) =$  Floating-Point  $(X_k)$   
99 Branch to P displaced by 2Q if Floating-Point  $(X_j) \neq$  Floating-Point  $(X_k)$   
9A Branch to P displaced by 2Q if Floating-Point  $(X_j) >$  Floating-Point  $(X_k)$   
9B Branch to P displaced by 2Q if Floating-Point  $(X_j) \geq$  Floating-Point  $(X_k)$   
9C Branch to P displaced by 2Q and increment  $(X_k)$  if  $(X_j) > (X_k)$   
9D Branch to P displaced by 2Q if segment  $(A_j) \neq$  segment  $(A_k)$   
9E Branch to P displaced by 2Q if exception per j in  $(X_k)$   
9F Branch to P displaced by 2Q and alter condition register per jk

### UNCONDITIONAL, INDEXED

2E Branch to P indexed by  $2(X_k)$  Right  
2F Branch to  $(A_j)$  indexed by  $2(X_k)$  Right

## IF FUNCTIONS

IF's two primary functions are accomplished by six major components. The first four components listed format and buffer incoming instructions. The last two produce instruction word addresses.

- Instruction Assembly.
- First Level Instruction Decoder.
- Instruction Formatting, Buffer Word Assembly.
- Instruction Buffer Ranks 2, 3, 10, 11, and 12.
- P Register.
- Branch Address Adder.

## INSTRUCTION ASSEMBLY

LM Read Data, an instruction word, enters IF after IF Request activates. Discussion on requirements for making IF Requests follows.

Disassembly of LM Read Data into individual instructions occurs in Instruction Assembly (IF 1.0). In either C170 or C180 mode, individual instructions are arranged within the instruction word on parcel (15 or 16 bit) boundaries. In either mode an instruction may be one or two parcels long. Refer to figures 3-1 and 3-2, which show various arrangements of instructions within instruction words. In C170 mode a two-parcel instruction is not permitted to cross a word boundary, while it is permitted in C180 mode.

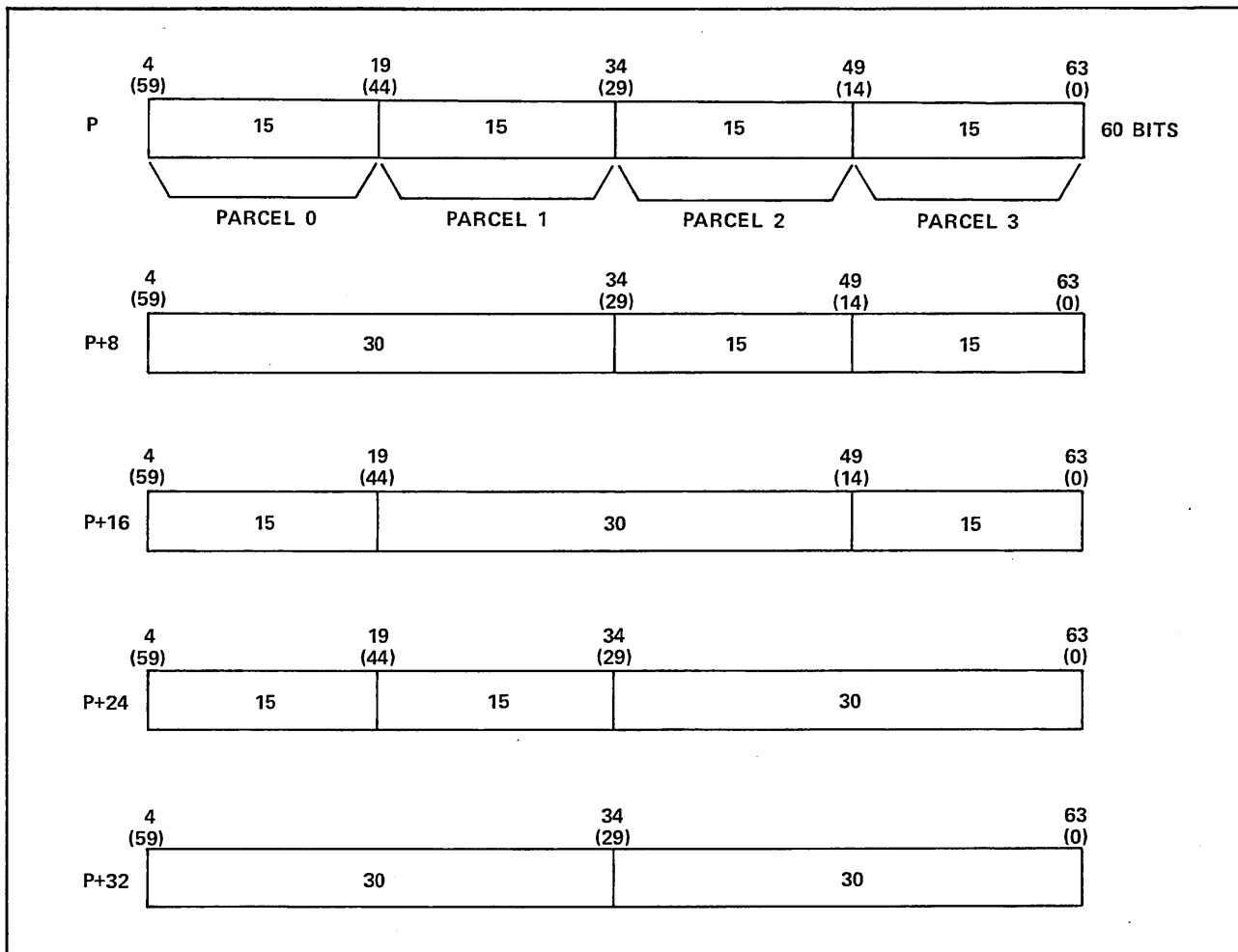


Figure 3-1. C170 Instruction Words

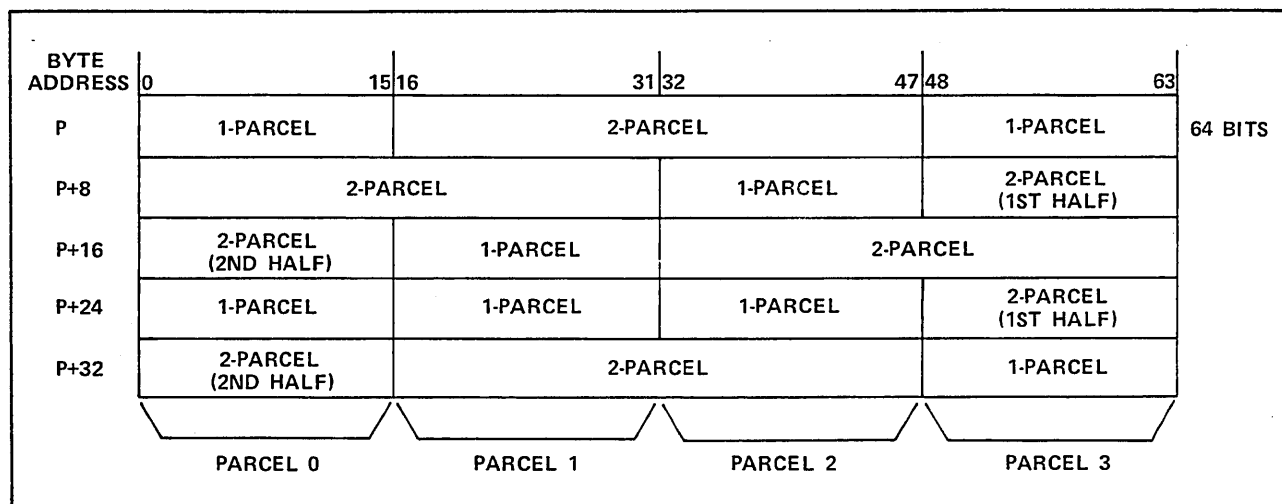


Figure 3-2. C180 Instruction Words

If the arriving instruction word is part of an RNI sequence, parcel selection begins with parcel 0. If in C180 mode the arriving instruction word is the first word of a branch or unbranch instruction sequence, the first selected instruction may be located in any parcel position. In C170 mode unbranching to parcel 2, 3, or 0 may occur, but branching is limited to parcel 0.

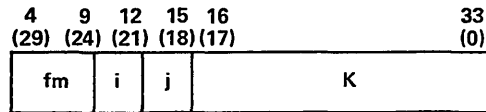
P Bits 61 and 62 control instruction selection in Instruction Assembly. These bits contain the parcel address of the instruction within the instruction word. They select two-parcel quantities, with the instruction left-justified in the event the instruction consists of a single parcel.

If during a C170 unbranch, C180 branch, or unbranch operation the desired instruction sequence starts in parcel 3, IF must submit two special case instruction word requests to memory. The first request obtains the instruction word containing the parcel 3 instruction. The second request is needed because the parcel count contained in the branch address cannot activate a normal RNI instruction word request. IF Request activates during an RNI sequence if parcel 1 is selected and the associated instruction is two parcels long or if parcel 2 is selected.

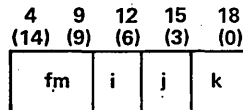
Whether an RNI, branch, or unbranch operation is taking place, parcel 3 always enters a holding register in Instruction Assembly when LM accepts the next instruction word request. In C180 mode a two-parcel instruction may cross a word boundary, requiring that parcel 3 of one instruction word be saved until the next instruction word arrives.

Formats of individual instructions selected in Instruction Assembly appear in figures 3-3, 3-4, and 3-5. During C180 BDP operations, one or two two-parcel BDP Descriptors follow the instruction parcel(s) through Instruction Execution hardware.

ijk INSTRUCTION FORMAT



ijk INSTRUCTION FORMAT



JK INSTRUCTION FORMAT

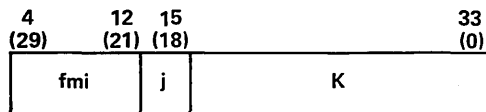
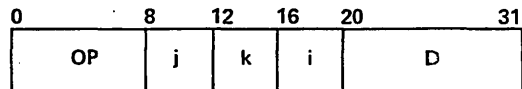
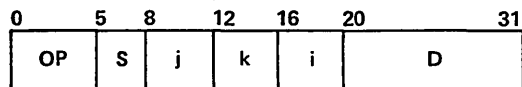


Figure 3-3. C170 Instruction Formats

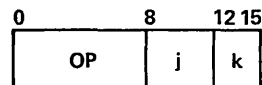
jkiD INSTRUCTION FORMAT



SjkiD INSTRUCTION FORMAT



jk INSTRUCTION FORMAT



jkQ INSTRUCTION FORMAT

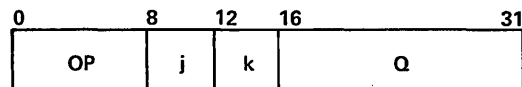
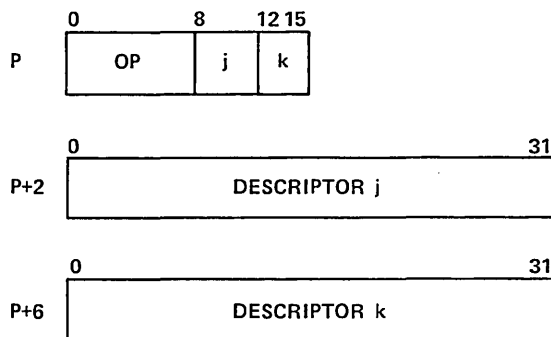


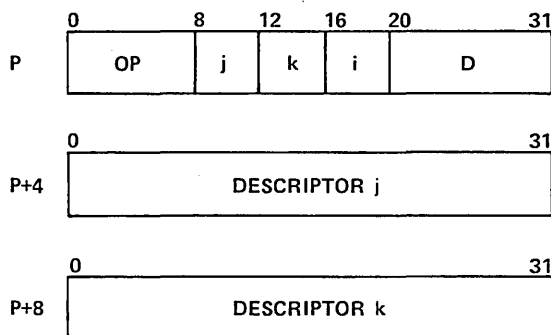
Figure 3-4. C180 Instruction Formats



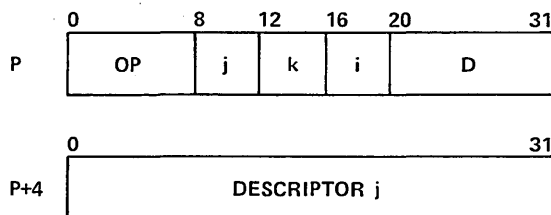
**jk PLUS TWO DESCRIPTORS**



**jkid PLUS TWO DESCRIPTORS**



**jkid PLUS ONE DESCRIPTOR**



**BDP DESCRIPTOR**

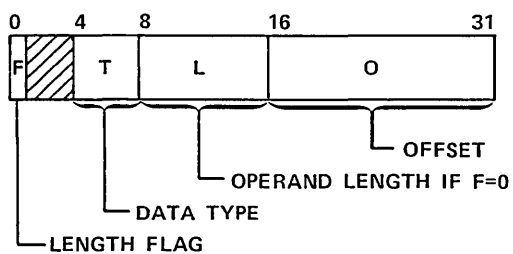


Figure 3-5. C180 BDP Instruction Formats

## FIRST LEVEL INSTRUCTION DECODER

The First Level Instruction Decoder contains supplemental information needed by an instruction for execution. The upper eight bits of the nine-bit C170 instruction opcode (fmi) or the eight-bit C180 Opcode address RAMs in the First Level Instruction Decoder. The RAMs' output data is in the formats shown in figure 3-6. First Level Instruction Decode Bits are defined in table 3-1.

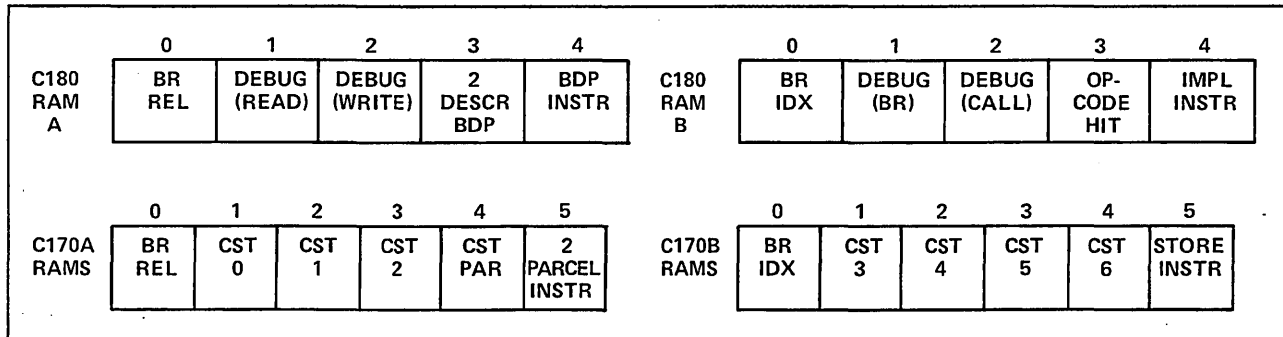


Figure 3-6. First Level Instruction Decoder Formats

Table 3-1. First Level Instruction Decode Bit Definitions

C170		
RAM	Bit	Definition
A	0	Indicates a relative branch instruction.
A	1-4	Indicates microcode address offset for CST RAM.
A	5	Indicates a 30-bit instruction.
B	0	Indicates an indexed branch instruction.
B	1-4	Indicates microcode address offset for CST RAM.
B	5	Indicates a C170 store instruction.
C180		
RAM	Bit	Definition
A	0	Indicates a relative branch instruction.
A	1	Indicates a read debug operation may occur in conjunction with instruction.
A	2	Indicates a write debug operation may occur in conjunction with this instruction.
A	3	Indicates a BDP instruction with two BDP Descriptors.
A	4	Indicates a BDP instruction.
B	0	Indicates a indexed branch instruction.
B	2	Indicates a call debug operation may occur in conjunction with instruction.

Two sets of C170 A, B RAM data proceed from the First Level Instruction Decoder to Instruction Formatting, Buffer Word Assembly (IF 1.0). The ninth opcode bit, contained in the C170 Parcel, selects the appropriate 12 bits. Only eight of the 10 C180 RAM A, B bits enter Instruction Formatting C180 Buffer Word Assembly. The other two, C180 RAM A Bits 3 and 4, enter control hardware in IF to handle the arrival of the BDP instruction and BDP descriptors in Instruction Formatting and Buffer Word Assembly. When a BDP Descriptor is selected in Instruction Assembly, the First Level Instruction Decoder output is not used.

#### INSTRUCTION FORMATTING, BUFFER WORD ASSEMBLY

Every C170 instruction, C180 instruction, or C180 BDP Descriptor is placed in a standard format in Instruction Formatting. Included in the reformatted instruction are field designators, CST RAM addresses and selected outputs from the First Level Instruction Decoder. Actual bit assignments are shown in figure 3-7.

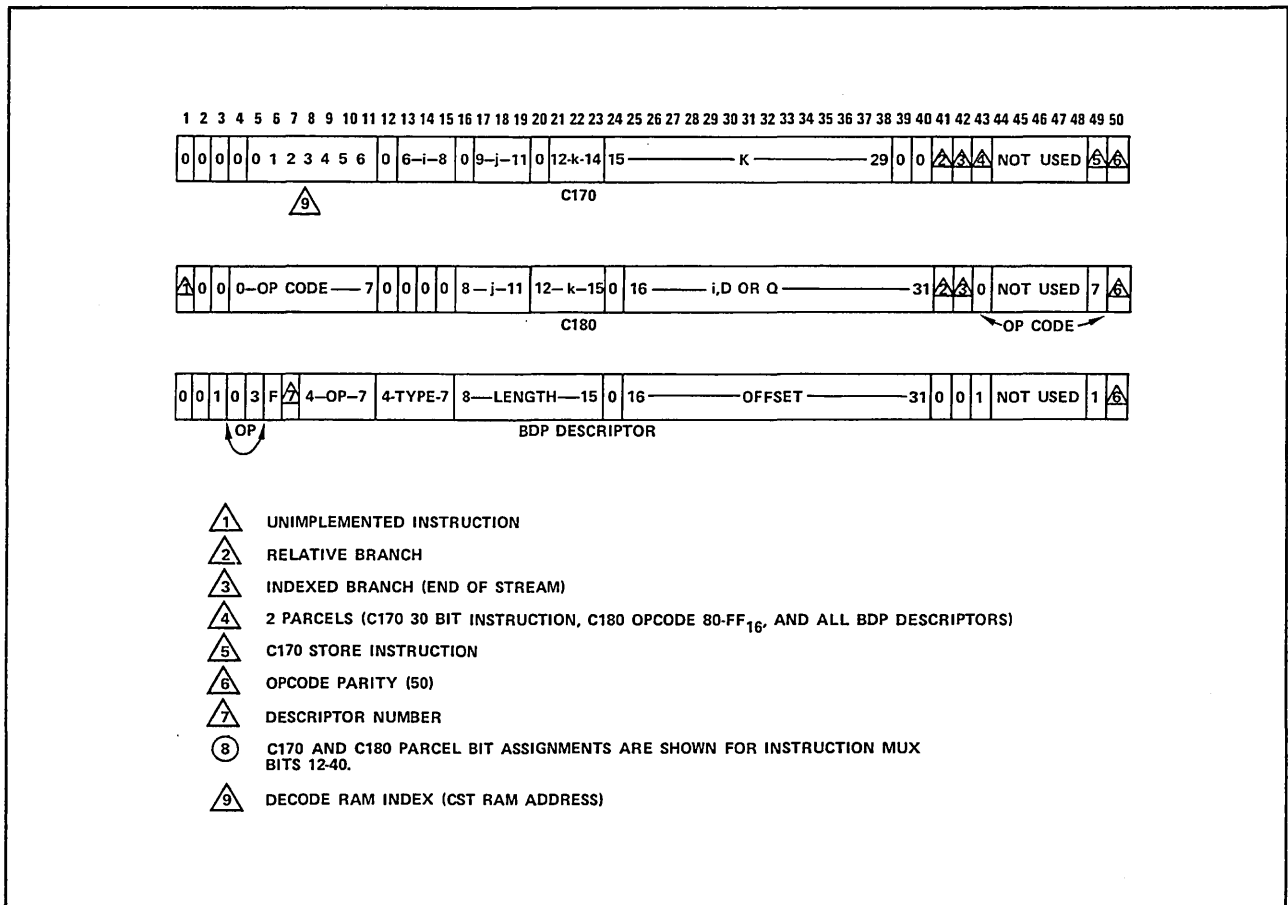


Figure 3-7. Instruction Formatting Bit Assignments

Additional data joins the output of Instruction Formatting in Buffer Word Assembly. Included are the current instruction address (P Bits 32 through 63) and selected C180 values from the First Level Instruction Decoder. The reformatted instruction proceeds to Instruction Buffer Rank 2 Register and Instruction Buffer Rank 12 Register. The Instruction enters rank 12 directly if ranks 3, 10, 11, and 12 are empty. In this case, rank 2 data is not used.

## INSTRUCTION BUFFER RANKS 2, 3, 10, 11, AND 12

The buffer ranks comprise the first stages of the instruction pipeline. The buffer ranks temporarily store a sequence of instructions before they are executed. After an instruction exits from ICP rank 22, each instruction in the Instruction Buffer advances one rank. Rank 12 Opcode enters a holding register in CST and addresses the appropriate sequence in the CST RAM. Rank 12 P and Rank 12 Instruction Mux data enter the Rank 13 Register in ICP.

Rank 2, 3, 10 through 12 Valid control movement in the Instruction Buffer. When active, the Valid signals latch data in the corresponding registers.

During an unconditional branch instruction, fetching of new instructions is suspended until the branch address is computed by ALN and becomes available at instruction Time 50. No new instructions enter the Instruction Buffer until after the address and accompanying IF Request are sent to LM, and LM Read Data returns.

If during a conditional branch instruction the condition is not met, ALN Unbranch activates. The Instruction Buffer is cleared of any instructions in the sequence that was branched to previously.

## P REGISTER

The P Register contains the PVA (a byte address) of the instruction selected in Instruction Assembly. That address (P) joins the instruction in Instruction Formatting, Buffer Word Assembly (IF 1.0) before the instruction enters Instruction Buffer Rank 2 or 12. At the same time the current instruction enters the Instruction Buffer Rank 2 Register, the byte address in the P Register increments by two or four (based on instruction size) and selects the next instruction in Instruction Assembly with P Bits 61 and 62. (The address increments by four if the previous instruction was two parcels long).

Branch Address enters the P Register as a result of a branching or an unbranching operation and designates a new instruction address.

## BRANCH ADDRESS ADDER

The Branch Address Adder is a 32-bit two's complement adder that calculates the address of the next instruction word to be requested from memory.

## RNI Instruction Request

The parcel count (P Bits 61 and 62) determines when the next instruction word is to be requested from LM. IF Request activates when the parcel count of the instruction selected in Instruction Assembly equals 2 (or 1 if the instruction is two parcels long). When the instruction which activates the request reaches Instruction Buffer Rank 2, it sends Rank 2 P to the P Mux and Parity Checker and Branch Address A Input Mux (IF 1.0). Branch Address A is then added to Branch Address B in the Branch Address Adder. Branch Address B is a literal 6 from Branch Address Network Control (IF 1.0). Because instruction words are addressed on word boundaries, the literal 6 guarantees the calculated address points to the next word in memory. IF Address proceeds to LM with IF Request. IF Address does not enter the P Register as Branch Address. Refer to figure 3-8 for timing information concerning the RNI instruction-word request sequence.

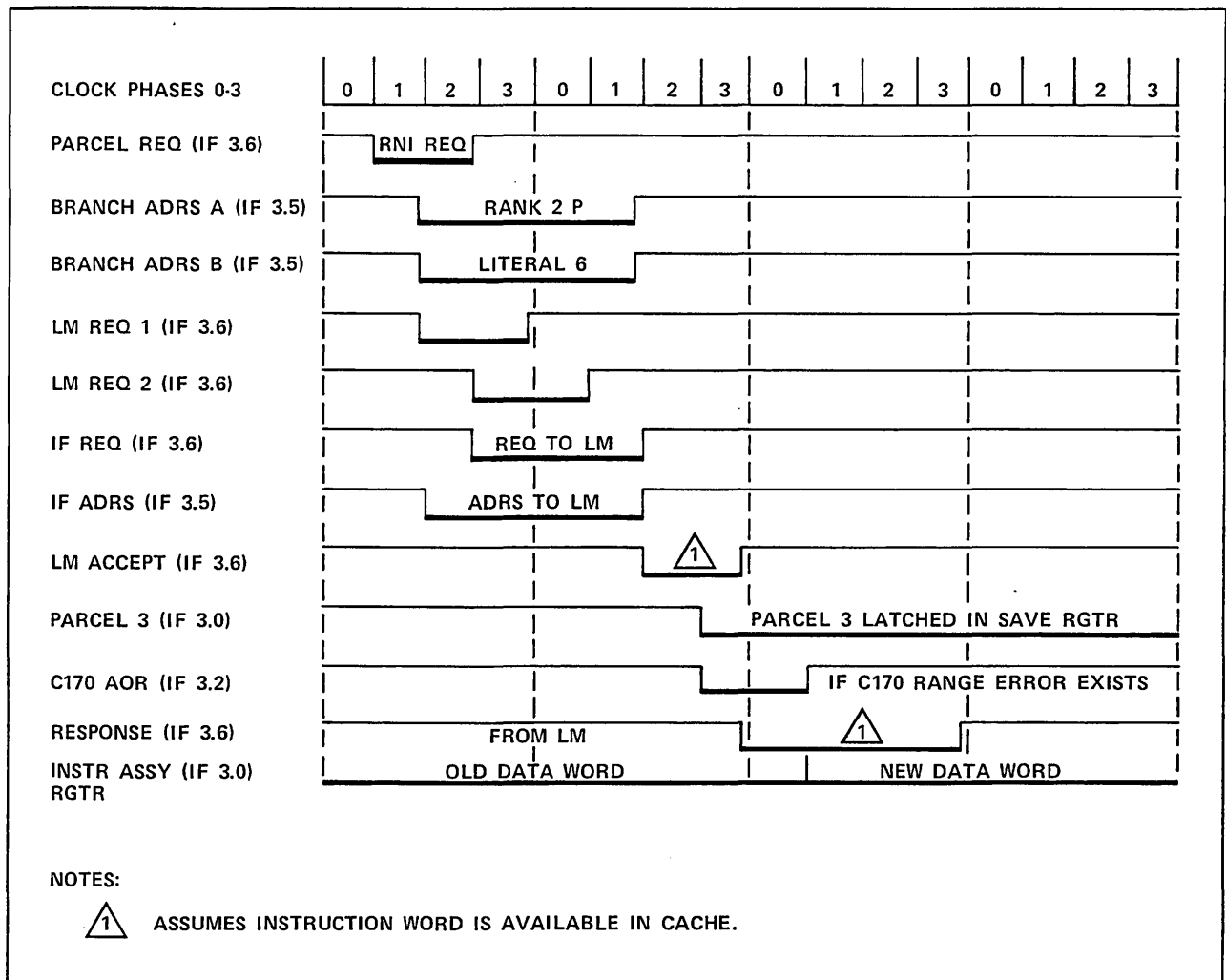


Figure 3-8. Instruction Fetch RNI

### Conditional Branch Instruction Request

Conditional branch instructions in C170 and C180 modes generate branch addresses in the IF Branch Address Adder.

In the case of a C170 conditional branch instruction, the P Mux selects eight times C170 RAC to become Branch Address A. Eight times Rank 2 C170 K enters the Branch Address B Adder and Input Mux to become Branch Address B. After these values are added in the Branch Address Adder, IF Address and IF Request proceed to LM. IF Address also enters the Branch Address Register. When LM acknowledges the request with IF Accept, Branch Address enters the P Register. The P Register selects the desired parcel from LM Read Data when the instruction word arrives from LM with IF Response. The steps are similar for a C180 conditional branch instruction except that IF Address is the sum of Rank 2 P and two times Rank 2 C180 Q. Refer to figures 3-9 and 3-10 for timing information and a flowchart describing a C180 conditional branch instruction.

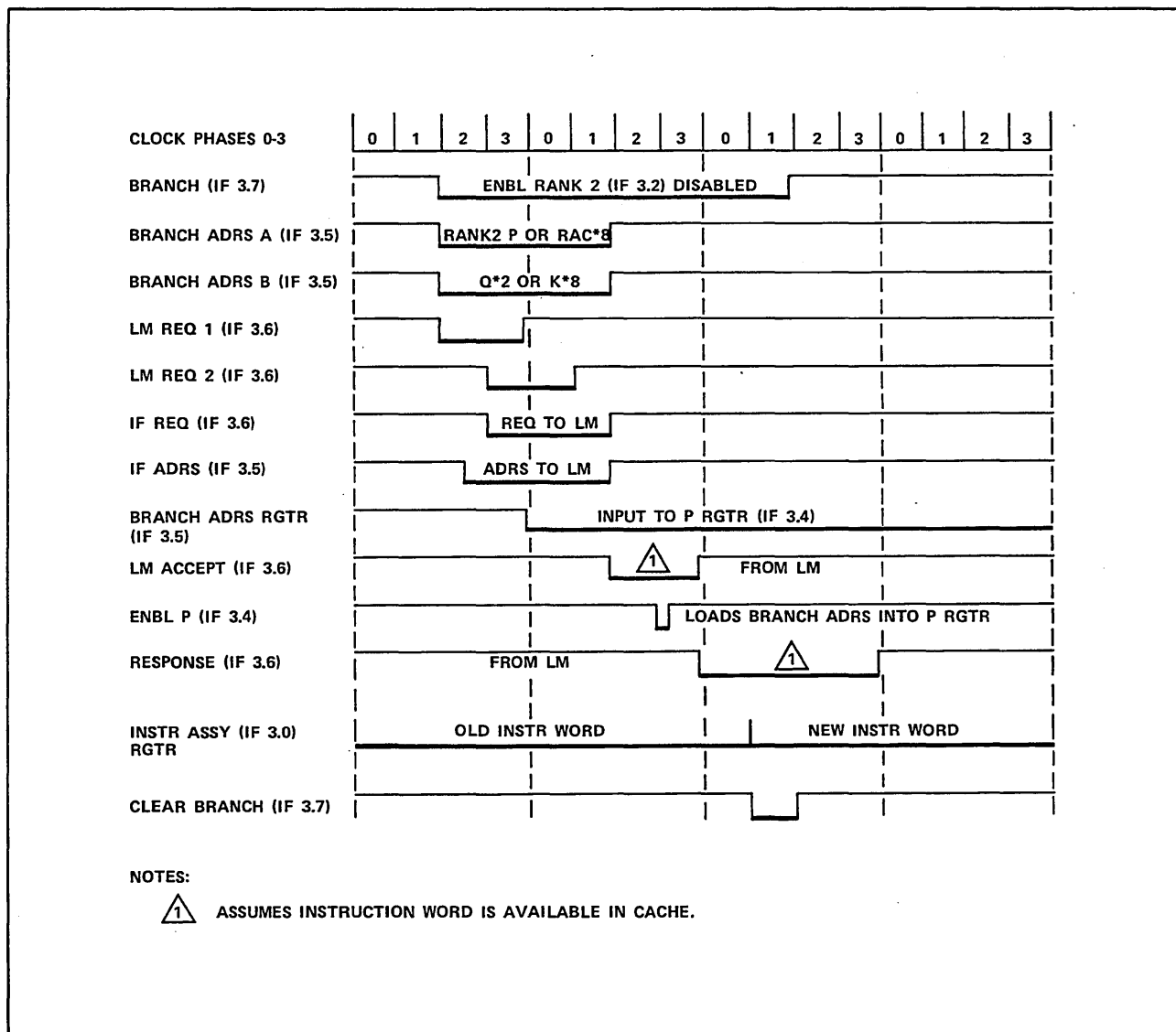


Figure 3-9. Conditional Branch (Not to Parcel 3)

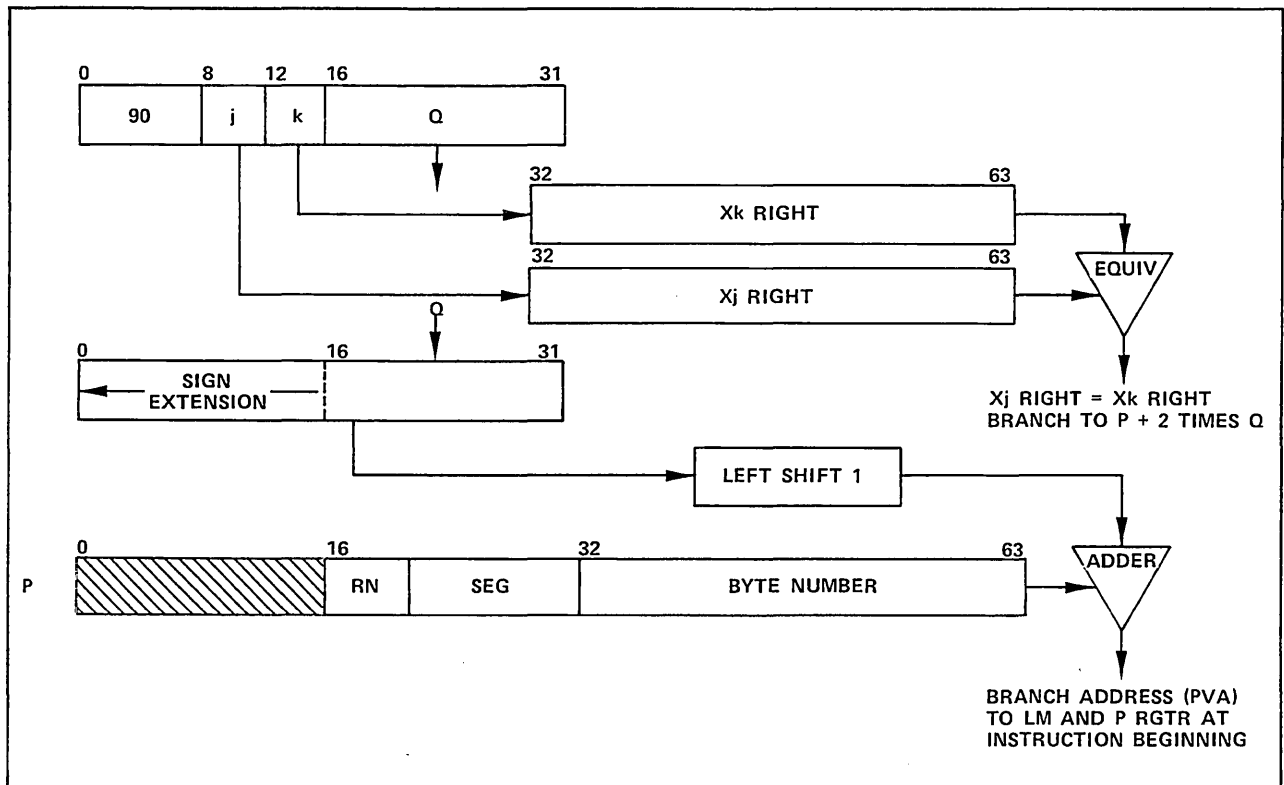
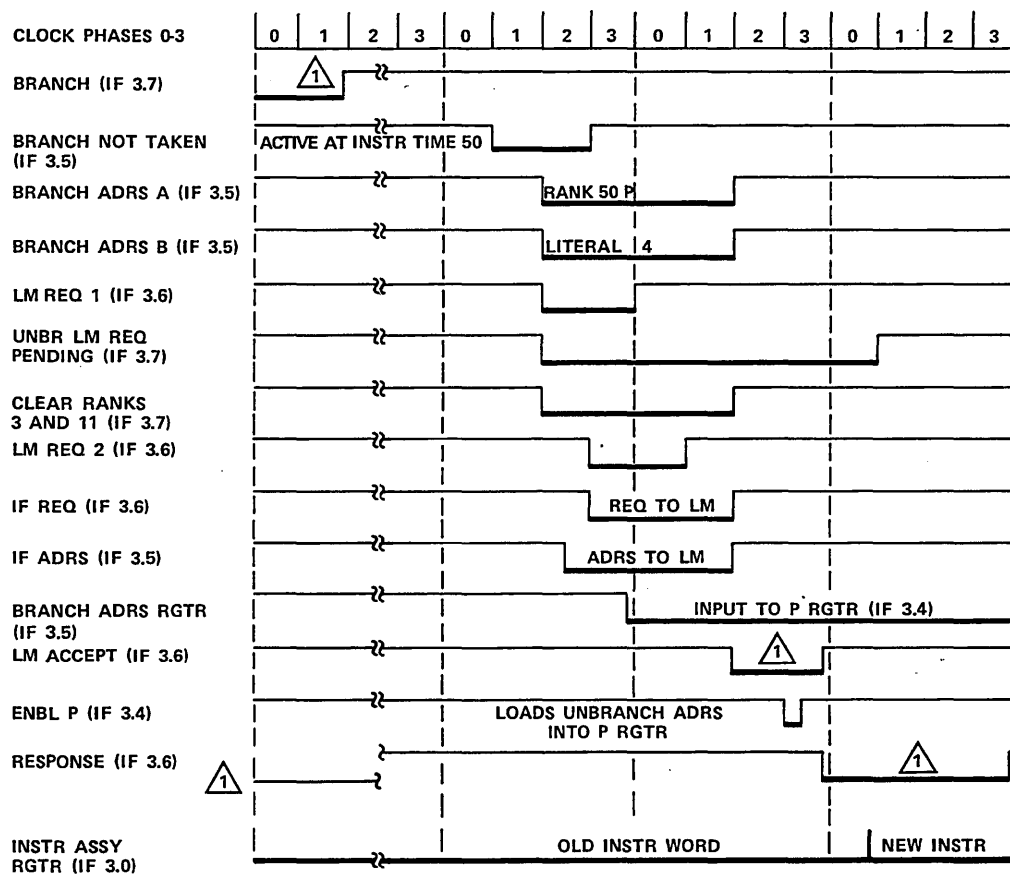


Figure 3-10. C180 (90) Branch to P Displaced by 2 Times Q If  $(X_j)$  Right =  $(X_k)$  Right

#### Unbranch Instruction Request

In the event the preset condition for a conditional branch instruction is not met, the instruction must unbranch by requesting the instruction following the branch instruction in the RNI routine. Because an unbranch condition is detected at branch instruction Time 50, Rank 50 P returns to IF from ICP to become Branch Address A. A literal 4 from Branch Address Network Control becomes Branch Address B. (Literal 4 is added to Rank 50 P because all conditional branch instructions are two parcels long.) IF Address, accompanied by IF Request, proceeds to LM. IF Address also enters the Branch Address Register. When IF Accept arrives, Branch Address enters the P Register. P selects the desired parcel in Instruction Assembly after LM Read Data and IF Response arrive from LM. Refer to figure 3-11 for details of the unbranch sequence timing.



NOTES:

1 ASSUMES INSTRUCTION WORD IS AVAILABLE IN CACHE.

Figure 3-11. Unbranch (Not to Parcel 3)

### Unconditional Branch Instruction Request

During unconditional branch instructions, ALN calculates the indexed branch address (New P) with operands from the Register File. New P from ALN (via OPI) enters the P Mux after instruction execution Time 50. (Branch Address B contains literal 0 from Branch Address Network Control.) New P becomes IF Address in the Branch Address Adder and proceeds to LM with IF Request. IF Address also enters the Branch Address Register for loading into the P Register when LM Accept arrives. P Bits 61 and 62 select the desired instruction parcel(s) when LM Read Data and IF Response arrive.



This address formation sequence also occurs during initialization, exchanges, instruction retries, calls, and other operations which require a new program address (P).

Refer to figures 3-12 and 3-13 for unconditional branch instruction timing information and a flowchart of indexed address formation for a C180 2F instruction.

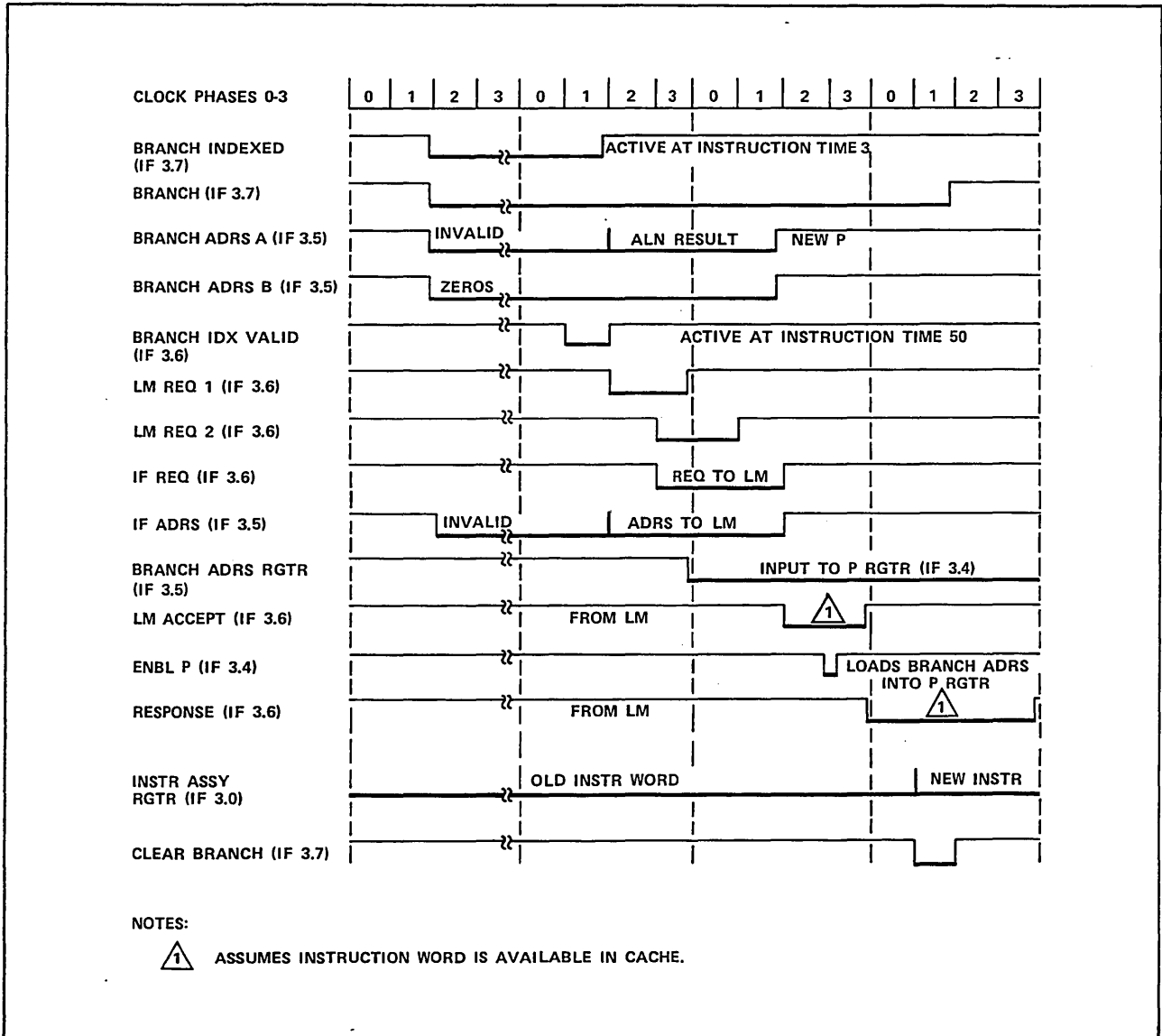


Figure 3-12. Unconditional Branch (Not to Parcel 3)

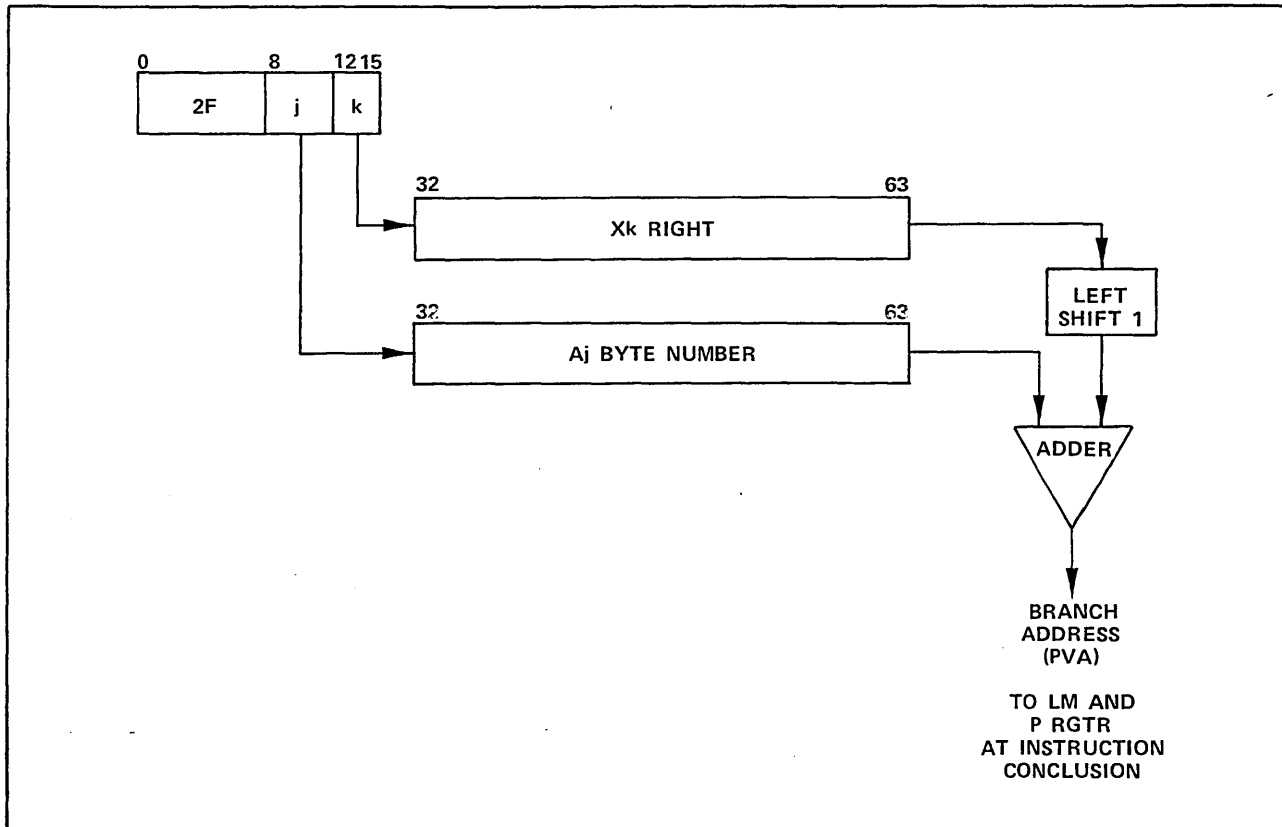


Figure 3-13. C180 (2F) Branch to P Displaced by 2 Times ( $X_k$ ) Right

#### Branch or Unbranch to Parcel 3 Instruction Word Request

If the branch address of an unbranch operation or C180 branch instruction is to parcel 3 of an instruction word, IF Requests obtain two instruction words. One instruction word contains the parcel 3 instruction. The next instruction word contains RNI sequence instructions and, if it is a two-parcel instruction, the second half of the parcel 3 instruction.

The branch address contained in the P Register enters the P Mux directly. Branch Address A is added to literal 6 (Branch Address B) to address the next instruction word. IF Address does not enter the P Register because normal incrementing of the original branch address controls selection of subsequent instructions.

Refer to figure 3-14 for timing information about a relative branch (to parcel 3) instruction.

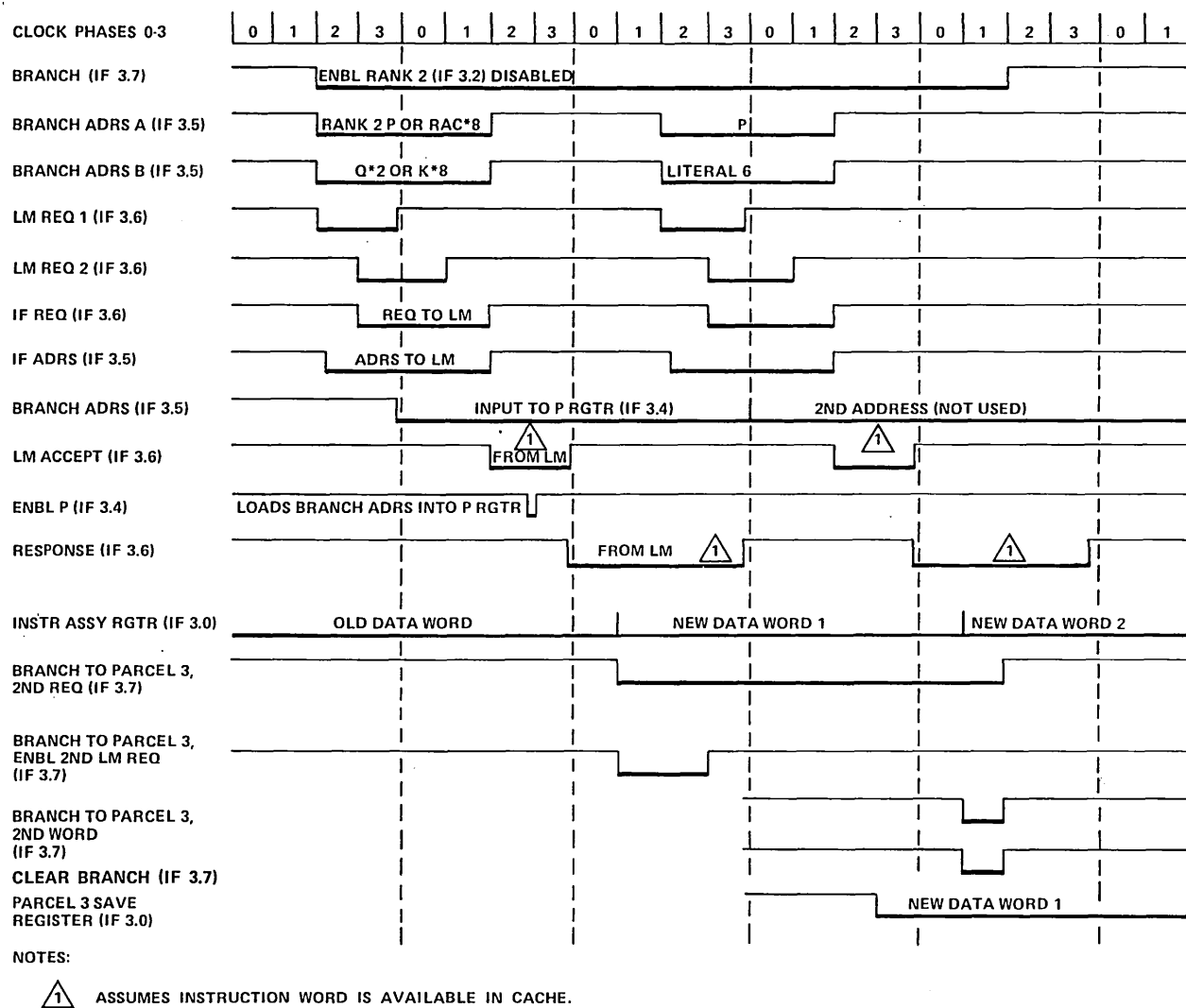


Figure 3-14. Conditional Branch (To Parcel 3)



## **PART 2**

### **INSTRUCTION EXECUTION**



ICP, CST, OPI, and ICC initiate and conclude execution of instructions IF obtains from memory. The role of ICC is reserved for discussion in section 6.

The CP's instruction buffering hardware, which pipelines instructions through IF and ICP, accommodates a maximum of 14 instructions/micrands. Included are as many as four unassembled instructions in IF. Ranks 2, 3, 10, 11, 12, and 13 in IF and ICP contain assembled instructions. The remaining ranks in ICP contain individual micrands associated with the instruction in the process of executing. An instruction's micrands advance through four execution stages in the ICP. One micrand enters a given stage after another departs.

If an instruction is already in the pipeline when its memory copy is changed, a self-modifying code error exists. Hardware provides the means to prevent execution of these unmodified instructions, but CP performance is reduced substantially when the feature is used.

A simplified picture of the coordinated operation of ICP, CST, OPI, and a typical executing functional unit is shown in figure 3-15.

The following sequence is based on signals and hardware shown in figure 3-15. A more detailed view of the hardware associated with instruction execution is available in the Level 3 diagrams referred to in figure 3-15.

1. Formatted instruction enters Instruction Buffer Rank 12 Register from Instruction Buffer Rank 2 or 11 if Rank 12 valid is inactive.
2. Rank 12 Valid activates to latch instruction in Rank 12.
3. At conclusion of micrand sequence of instruction currently in Rank 22, Micrand Sequence Control generates exit. Last micrand enters General Micrand Register and FU Micrand Buffer Register.  
  
Contents of Rank 12.5 Register enter Micrand Address Mux Register. Register contains first CST RAM address associated with instruction in Rank 13.
4. Exit gates Rank 12 Opcode into Rank 12.5 Address Register and partially enables instruction in Rank 13 to Rank 22.
5. Second-to-last General Micrand of Rank 22 instruction activates FU Go while in Rank 32 and initiates micrand-controlled execution in Functional Unit. FU Go activates, provided response signal associated with previous micrand clears Wait For Response FF and Functional Unit the micrand needs to use is not busy. FU Go enters selected functional unit to latch Functional Unit Micrand into holding register.
6. FU Go gates General Micrand from Rank 32 to Rank 41, activates FU Advance, and sets Wait For Response FF.
7. FU Advance generates Advance Pipeline and Next Micrand, provided micrand entering Rank 41 does not have operand conflict. (Movement of micrands in pipeline may be delayed until required operands are available to Functional Unit receiving FU Go.)  
  
FU Advance gates last FU Micrand of Rank 22 instruction to FU Micrand Register from FU Micrand Buffer Register.
8. Advance Pipe and CST Rank 22 Valid (a derivative of Next Micrand) gate final General Micrand of Rank 22 instruction to Rank 32. Rank 22 instruction fields enter Rank 32 for last time (these fields accompany all micrands of instruction through pipeline).

Advance Pipe and Exit gate next instruction from Rank 13 to Rank 22.

First micrand associated with new instruction entering Rank 22 enters General Micrand Register and FU Micrand Buffer Register.

Micrand Sequence Control field from same micrand enters Micrand Address Mux Register to select next micrand from CST RAM.

9. Operand read addresses (RDSa, b, c) for first micrand of new instruction begin Register File access. Read addresses also are compared to contents of Minipipe delay network to detect possible conflicts.

Register File write address (RSDw) for first micrand of new instruction enters Minipipe. Delay circuit synchronizes entry of write address into Register File with arrival of result.

10. Rank 12 Valid and inactive Rank 13 Valid gate instruction from Rank 12 to Rank 13.
11. Instruction from Instruction Buffer Rank 2 or 11 enters Instruction Buffer Rank 12.
12. Final General Micrand of first instruction activates FU Go.
13. Operation associated with FU Go begins in Functional Unit, employing operands from Register File RAMs under Functional Unit Micrand Control.
14. Steps 6 and 7 repeat. (Variation - first FU Micrand of new Rank 22 instruction enters FU Micrand Register in step 7.)
15. Advance Pipe and CST Rank 22 Valid gate first General Micrand of new instruction to Rank 32 along with new P and Instruction Control Signals.

Second micrand associated with new instruction in Rank 22 enters General Micrand Register and Functional Unit Micrand Buffer Register.

Second Micrand Sequence Control field associated with new instruction in Rank 22 enters Micrand Address Register to select third micrand from CST RAM.

16. Step 9 repeats (with fields from second General Micrand).
17. Result of operation associated with last micrand of first instruction proceeds to location in Register File addressed by RDSw.
18. First micrand of new Rank 22 instruction activates FU Go, provided Response associated with previous micrand has cleared Wait For Response FF and Functional Unit the micrand intends to use is not busy.
19. Operation associated with FU Go begins in Functional Unit, employing operands from Register File RAMs under Functional Unit Micrand Control.
20. Steps 6 and 7 repeat. In step 6 new P and Instruction Control Signals enter Rank 44 with General Micrand. In step 7 second FU Micrand enters FU Micrand Register.
21. Advance Pipe and CST Rank 22 Valid gate second General Micrand of Rank 22 instruction to Rank 32.

Third micrand associated with Rank 22 instruction enters General Micrand Register and Functional Unit Micrand Buffer Register.



Third Micrand Sequence Control field associated with Rank 22 instruction enters Micrand Address Register to select fourth Micrand from CST RAM.

22. Step 9 repeats (with fields from third General Micrand).
23. Result of first micrand operation associated with Rank 22 instruction proceeds to Register File. Register File is addressed by RDSw (which entered Minipipe in step 9).

Steps 1 through 23 repeat until the instruction sequence is exhausted or an interrupt occurs. Traps, calls, and exchanges are typical means of ending one instruction sequence and starting another.

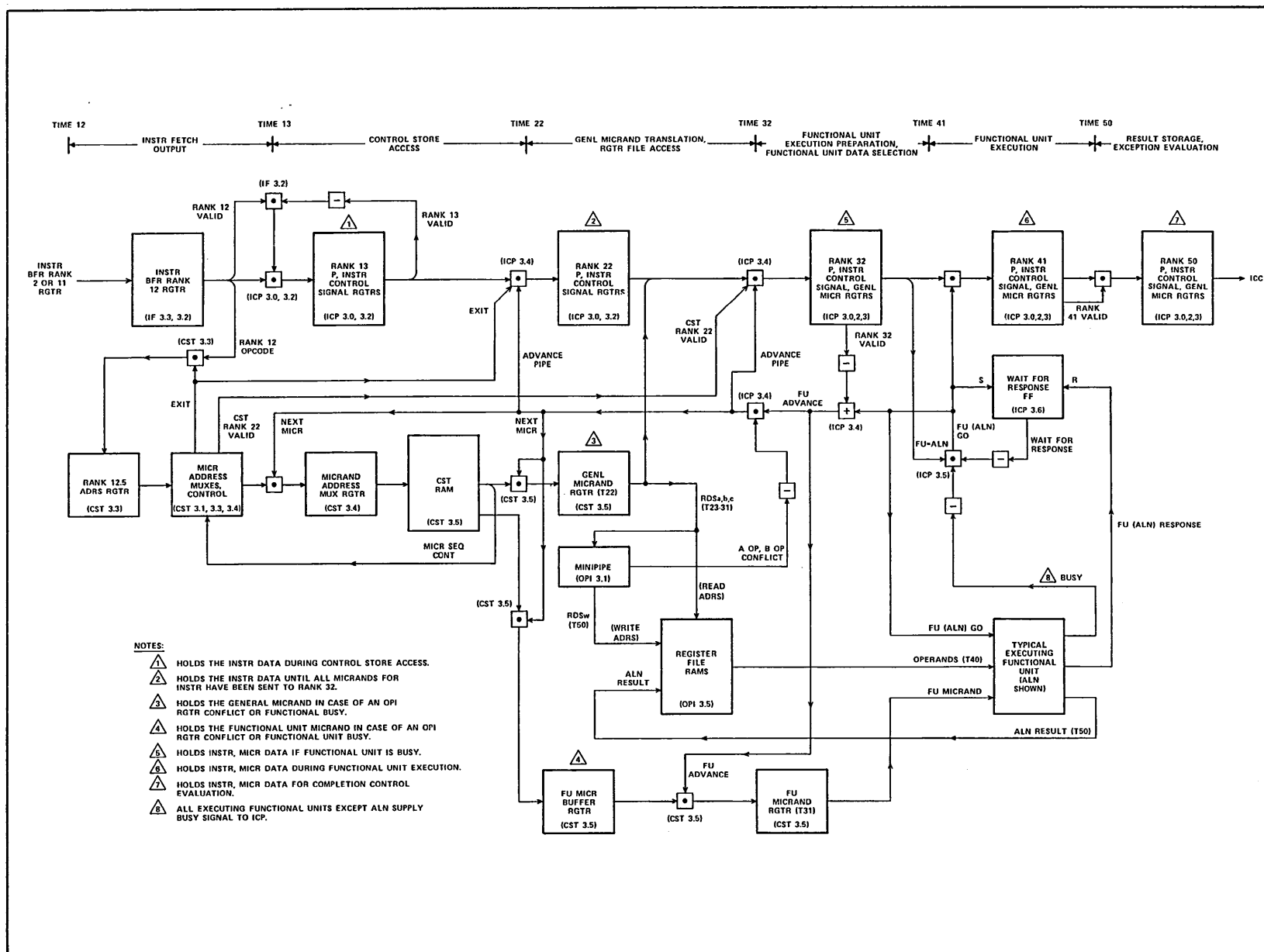


Figure 3-15. Instruction Unit Operation

### CONTROL STORE (CST 1.0)

CST stores and distributes micrands to provide microcode control of ICP, OPI, AC, ALN, BDP, SM, and LM, as well as some of its own operations. OPI, ICP, and CST receive portions of the General Micrand. ALN and BDP each receive the entire Functional Unit Micrand. AC, SM, and LM each receive a portion of the Functional Unit Micrand.

During MAC operations, CST disassembles 64-bit data words from the Register File or the Functional Unit Micrand Register and delivers eight-bit bytes to the MAC Data Bus.

CST also contains logic to start or stop the CP and to determine breakpoints within instruction sequences. When DEC Bit 44 is active, a match between the contents of the Breakpoint Register (CST 3.3) and the CST address causes CST RAM addressing to halt. A halt permits evaluation of CP operating register values that exist when the match occurs. When DEC Bit 44 is inactive, the breakpoint compare circuitry may be used to trigger scoping operations in a micrand routine beginning at the point of the match.

### CST RAM

The CST RAM of models AD112-C and AD113-A contains 4,096 memory locations, each of which accommodates one 128-bit microcode word plus 16 parity bits (one parity bit for each data byte). The contents of Control Store RAM are treated as two separate memories, in effect creating 2,048 pairs of 128-bit words. The second memory is called the shadow memory. It acts as a duplicate, back-up memory. In the event of a parity error, a toggle to the shadow memory should eliminate the need to place Control Store offline. However, if the parity error occurs among the micrand sequence control bits 0-15, the two leftmost bytes, a standard interrupt will occur.

Models AD112-A, B do not have the shadow memory feature. These models have a CST RAM with 2,048 memory locations, each of which accommodates one 128-bit microcode word.

MAC writes the CST RAM during system initialization via the Maintenance Channel. Once the CP is placed online by a Start operation, MAC can no longer write the CST RAM. Refer to section 2, IOU Read, Write Operations for details of the CST RAM write sequence.

The General Micrand comprises 64 bits of the 128-bit microcode word. The remaining 64 bits are the Functional Unit Micrand. Primary General Micrand operations include Functional Unit selection, Micrand sequence control, operand selection, and result storage. The Functional Unit Micrand controls executing Functional Units. Select Functional Unit Micrand from Micrand Addressing and Sequence Control controls output of the two 64-bit quantities from the CST RAM location addressed by CST Address.

### MICRAND ADDRESSING AND SEQUENCE CONTROL (MSC)

Micrand Addressing and Sequence Control determine the address of the microcode word to be read or written in the CST RAM. The address of the first micrand in an instruction or other micrand sequence comes from an external source. Addresses for subsequent micrands in that sequence come from the preceding General Micrand's MSC field or from a RAM address incrementer. The MSC field also determines when the instruction or micrand sequence in ICP Rank 22 exits.

Because the General Micrand and the Functional Unit Micrand use a time-shared path to enter the General Micrand or Functional Unit Micrand registers, Select Functional Unit Micrand determines when either may use that path. When routine online CST RAM reads occur, clocks manipulate Select Functional Unit Micrand to allow both micrands from one CST RAM location to travel the data path during one major cycle. Because the General and Functional Unit micrands must be disassembled into bytes when MAC is reading the CST RAM offline, MAC controls when Select Functional Unit Micrand is active or inactive. Refer to section 2, IOU Read, Write Operations for details of the offline CST RAM read sequence.

### Entry Addresses

Typically, the initial CST RAM address for a given micrand sequence comes from one of the following sources.

- Rank 12 Opcode.
- MAC Microcode Address.
- Reference ROM data.
- Microtrap Code.

#### Rank 12 Opcode

Rank 12 Opcode contains the microcode address index of the instruction occupying the Instruction Buffer Rank 12 Register (IF 1.0). In Micrand Addressing and Sequence Control, the opcode value is added to a constant of  $300_{16}$  to address a C170 instruction entry point. The constant is  $100_{16}$  for a C180 instruction entry point. When a C180 BDP Descriptor occupies rank 12, a constant of  $200_{16}$  is added to the saved BDP instruction opcode, descriptor number, and descriptor length flag.

#### MAC Microcode Address

The IOU may read or write a CP-resident register to which it does not have direct access. MAC supplies a constant to address the CST RAM entry point of the micrand sequence the CP needs to perform the operation on the IOU's behalf. The MAC Microcode Address contains the CST RAM entry point address.

Before MAC can read or write the CST RAM offline, MAC must load the initial RAM address into the micrand address register (MAR). The IOU sends the address to CST via the MAC Microcode Address path. The IOU uses the same path to write the breakpoint register in Micrand Addressing and Sequence Control. Refer to section 2, IOU Read, Write Operations for details of MAR and breakpoint register write sequences.

#### Reference ROM Data

At an intermediate stage in IOU microcode-assisted read or write operations, a new micrand address is formed from the contents of the Reference ROM in MAC addressed by MAC Address and a constant value. The micrand sequence beginning at the calculated address performs the actual read or write of the designated CP-resident register.

The CP uses Reference ROM data for the same purpose when performing C180 Copy State (OE, OF) instructions. The contents of the X<sub>j</sub> Register address the Reference ROM. When a copy state instruction reads or writes a register which the IOU also requires microcode assistance to enter, the instruction addresses the same microcode sequences in the CST RAM using the same Reference ROM indexes. Refer to section 2, IOU and CP Read, Write Operations for details of IOU and CP microcode-assisted sequences.

#### Microtrap Code

The CST RAM entry points for ICC-initiated interrupt routines are produced in CST from the Microtrap Code. The entry points and associated operations are:

- 040<sub>16</sub> Instruction retry with purge.
- 041 Instruction retry.
- 042 Trap with purge.
- 043 Trap.
- 044 C180 exchange with purge.
- 045 C180 exchange.
- 046 C170 exchange.
- 047 Halt.

Refer to section 6 for details of interrupt routines initiated by ICC.

#### Other CST RAM Entry Addresses

In between instruction sequences, CST normally starts an interrupt routine at one of the following entry points.

- 010 IF Rank 12 not valid (unconditional exit).
- 020 IF Rank 12 not valid (conditional exit).

When abnormal conditions are indicated by IF or when the IOU stops the CP, interrupt routines start at the following CST RAM entry points.

- 004<sub>16</sub> Stop (initiated by IOU).
- 008 Unimplemented instruction.
- 380 Unexecutable instruction.

### CST Opcode

When MAC reads or writes the CST RAM, the micrand address register, or breakpoint register, MAC controls the operations with CST Opcode. The CST Opcode transfers control of Select Functional Unit Micrand to MAC for reading the CST RAM. During a CST RAM write, the CST Opcode allows incremented addresses to enter the CST RAM. Because the breakpoint and micrand address registers share common paths to and from MAC, CST Opcode designates which is to be read or written.

### GENERAL MICRAND REGISTER

The least significant 48 bits of the General Micrand enter the General Micrand Register. The functional unit and General Micrand extension fields proceed to ICP, along with point of no return and clock condition register bits. Counter control, functional unit, write data select and RDSa, b, and c fields enter OPI.

### FUNCTIONAL UNIT MICRAND REGISTERS

Two registers accommodate two consecutive Functional Unit Micrands. The registers are used to synchronize the micrands with associated General Micrand fields in CST's General Micrand Register and the ICP Rank 32 General Micrand Register.

For performance sake, the Functional Unit Micrand proceeds to all Functional Units. Only the Functional Unit receiving a Go signal uses the Functional Unit Micrand.

### DISASSEMBLY NETWORK

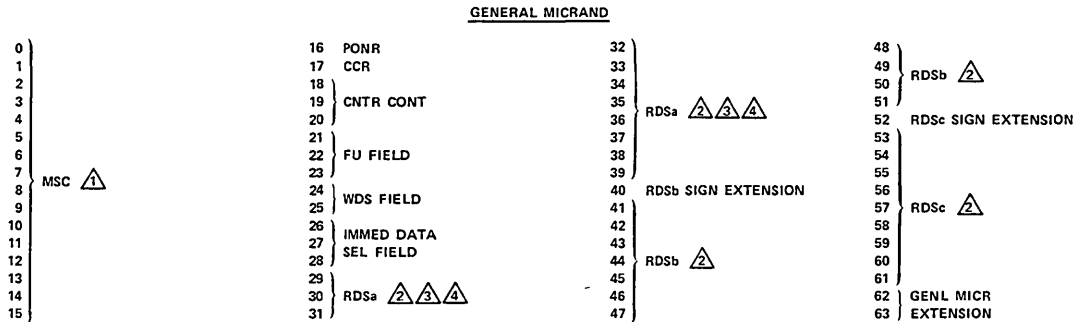
The Functional Unit Micrand path permits the Functional Unit Micrand or the General Micrand to enter the Disassembly Network during MAC read operations. MAC controls disassembly and obtains data bytes via the MAC Data Bus.

MAC also controls the disassembly network when reading the Register File. PFS Data Bus Output proceeds unmodified through the disassembly network when read by either the IOU or the CP. Refer to section 2, IOU Read, Write Operation for details of Disassembly Network use during CST RAM, Register File, and PFS read operations.

### MICRAND FORMATS

Figures 3-16 and 3-17 show the General Micrand and Functional Unit Micrand formats. Additional text describes the uses of the various bits and fields within the micrands.

BYTE 0								BYTE 1								BYTE 2								BYTE 3								BYTE 4								BYTE 5								BYTE 6								BYTE 7							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63



NOTES:

$\Delta$  THE MSC FIELD HAS THREE FORMATS:

BIT															
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
CB=0															
BA															
LRT CB MISC															
CB=1, CA=0															
CA CAE CD															
LRT CB MISC															
CB=1, CA=1															
CA CS BD															
LRT CB MISC															

$\Delta$  THE RDSa, b, AND c FIELDS HAVE FIVE READ OPERATION FORMATS:

	BIT															
RDSa	29	30	31	32	33	34	35	36	37	38	39					
RDSb	41	42	43	44	45	46	47	48	49	50	51					
RDSc	53	54	55	56	57	58	59	60	61							
RD=0,LT=0	RD	LT	BI	OR	RAF			UZ	DS							
RD=0,LT=1	RD	LT	BI	LRA			UZ	DS								
RD=1,LO=0	RD	LO	LD						DS							
RD=1,LO=1,DL=0	RD	LO	DL	DSS			DS									
RD=1,LO=1,DL=1	RD	LO	DL	LRR			DS									

$\Delta$  THE RDSa AND d FIELDS HAVE THREE WRITE OPERATION FORMATS:  
(RDSd IS PART OF THE AC FUNCTIONAL UNIT MICRAND.)

BIT											
RDSa	29	30	31	32	33	34	35	36	37	38	39
RDSd	53	54	55	56	57	58	59	60	61	62	63
RD=0, LT=0	RD	LT	BI	OR	RAF				WW		
RD=0, LT=1	RD	LT	BI	LRA				WW			
RD=1	RD	LRW/CSC									

$\Delta$  RDSa MAY BE EITHER A WRITE DESTINATION OR CM ADDRESS ARGUMENT. FOR ALN AND BDP OPERATIONS IT IS A WRITE ADDRESS. FOR AC OPERATIONS IT IS AN ADDRESS ARGUMENT.

Figure 3-16. General Micrand

BYTE 0								BYTE 1								BYTE 2								BYTE 3								BYTE 4								BYTE 5								BYTE 6								BYTE 7							
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63

0

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

PARTIAL SC POINTER

MEM WRITE

ADRS RESTRICTION

BYTE ASSY/DISASSY

LEFT TO RIGHT ADDRESSING

ADRS MODULE DESIGNATOR

ADRS LENGTH

ADRS SPEC TEST

PAGE VALID

FCTN CODE

ADRS OUT OF RANGE

AC

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

NOT USED

PURGE CACHE

NOT USED

STR MODE

CMC FCTN CODE

RETURN PAGE TABLE INDEX

TRANSLATE SVA TO RMA

SUPPRESS PSWF

RESPONSE DIRECTOR

INITIALIZE PAGE TABLE

CACHE READ

LM

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

PURGE PER CODE

EXPLICIT PURGE CODE

RMA

SVA

INVALID SEG

RING AND SEG SEL

ALLOCATE

READ VALIDATE

WRITE VALIDATE

NOT USED

FCTN CODE

CACHE BYPASS

SM

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

NOT USED

ZERO FILL

RDSd SIGN EXTENSION

RDSd

(REFER TO GENL MICR SHOWN IN FIGURE 3-16, NOTE 3.)

0

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

ERROR TEST GROUP

FORCE DATA TYPE

FORCE A<sub>j</sub> TYPE INP

FORCE A<sub>k</sub> TYPE INP

FORCE A<sub>k</sub> TYPE OUT

LOAD MSNZB

RESET BFR RAM ADRS CNTR

FILL BYTE

GATE A STR

GATE B STR

DEC ALU

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

LWR DIGIT SUBTR

INP TERMN

COND SEL

MISCOMPARE TERMN

NO OPN IF A<sub>k</sub> LENGTH=0

DSBL PDM

NOT USED

INP SEL

OUT REQUIRED

UPR DIGIT SUBTR

LEFT TO RIGHT

IMMEDIATE

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

IMMEDIATE FIRST BYTE

CLR SIGNS

ENBL ARITH OVFL

ENBL ARITH LOS

ENBL BDP DIV FAULT

PROPAGATE 4-BIT CARRY

PROPAGATE 8-BIT CARRY

INH LWR TO UPR DIGIT CARRY

FCTN A

FCTN B

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

BFR RAM INP SEL

A STR INP SEL

TABLE LD TERMN

WAIT FOR A<sub>k</sub> DESCR

CALCULATE SUBSCRIPT

CLR BUSY

NOT USED

PONR

2'S COMP CONT

FORCE +1 IF A<sub>j</sub> NEG

MSNZB

0

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

CONSTANT

RESPONSE

SHIFT CNT

GEN FIELD

MUX CONT

FIELD

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

COMP FIELD

EXP ARITH

FIELD

NORM

ENCODE

CONT

LARGE ADDER CONT

(BITS 64-95)

LARGE ADDER CONT

(BITS 32-63)

LARGE ADDER

CARRY CONT

32

33

34

35

36

37

38

39

40

41

42

43

44

45

46

47

LARGE ADDER CARRY CONT

PLUS 1

ALN OUT MUX CONT

(BITS 0-15)

ALN OUT MUX CONT

(BITS 16-63)

DOUBLE PRECISION

ENBL EXP UNDFL, OVFL

EXCEPTION SEL

FIELD

C170 MODE

ENBL ENDCASE COND

ENDCASE/BR COND

SEL FIELD

48

49

50

51

52

53

54

55

56

57

58

59

60

61

62

63

ENBL FLOATING POINT COMPARE

SHIFT CNT SEL

ROUND CONT

MULT/DIV START

SOFT CONT

ENTRY ADRS

SHIFT NET ZERO EXTEND

MULT/DIV RUN

ENBL 18-BIT COMPARE

FORCE ZERO EXP



## General Micrand

The General Micrand controls the micrand sequence, read and write operand data paths, functional unit selection, and Register File address selections. The General Micrand is shown in figure 3-16 and described in table 3-2.

Table 3-2. General Micrand Bits and Descriptions (Sheet 1 of 11)

Bit	Name	Description																
0-15	Micrand Sequence Control (MSC)	<p>Determines next micrand address to enter CST RAM (CST 3.5). Address selection is accomplished by control subfields. Control subfields include Condition Action (CA), Conditional Branch (CB), and Condition Action Extension (CAE). The Branch Delta (BD), Branch Address (BA) and Condition Data (CD) subfields may be used in address calculations. Load Return Register (LRT) and Miscellaneous (MISC) subfields initiate special operations.</p> <p>If CB=0, BA value contained in MSC Bits 0-10 (CST 3.4) permits branching to any CST RAM location.</p> <p>If CB=1 and CA=0, CAE subfield (CST 3.1) determines operation. Operation may require use of CD value contained in MSC Bits 6-10 (CST 3.4).</p> <table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0 or 4</td><td>Unconditional exit - Used between instruction and first descriptor, as well as between two descriptors during BDP instruction execution.</td></tr><tr><td>1 or 5</td><td>Micrand sequence branches to CD subfield X 100<sub>16</sub> + Reference ROM data. Used in MAC microcode-assisted and CP copy state instruction operations to convert register designator into CST RAM entry address.</td></tr><tr><td>2</td><td>If Debug condition exists (CST 3.1) and Debug Lockout is inactive (CST 3.1), micrand sequence branches to 080<sub>16</sub> + CD. Debug Lockout activates.</td></tr><tr><td></td><td>If conditions are not met, CST RAM address increments.</td></tr><tr><td>3</td><td>If Debug condition exists and Debug Lockout is inactive, micrand sequence branches to 0C0<sub>16</sub> + CD. Debug Lockout activates.</td></tr><tr><td></td><td>If Debug condition exists and Debug Lockout is active, micrand sequence branches to 7FF<sub>16</sub>.</td></tr><tr><td></td><td>If neither case applies, exit occurs.</td></tr></table>	<u>Decode</u>	<u>Description</u>	0 or 4	Unconditional exit - Used between instruction and first descriptor, as well as between two descriptors during BDP instruction execution.	1 or 5	Micrand sequence branches to CD subfield X 100 <sub>16</sub> + Reference ROM data. Used in MAC microcode-assisted and CP copy state instruction operations to convert register designator into CST RAM entry address.	2	If Debug condition exists (CST 3.1) and Debug Lockout is inactive (CST 3.1), micrand sequence branches to 080 <sub>16</sub> + CD. Debug Lockout activates.		If conditions are not met, CST RAM address increments.	3	If Debug condition exists and Debug Lockout is inactive, micrand sequence branches to 0C0 <sub>16</sub> + CD. Debug Lockout activates.		If Debug condition exists and Debug Lockout is active, micrand sequence branches to 7FF <sub>16</sub> .		If neither case applies, exit occurs.
<u>Decode</u>	<u>Description</u>																	
0 or 4	Unconditional exit - Used between instruction and first descriptor, as well as between two descriptors during BDP instruction execution.																	
1 or 5	Micrand sequence branches to CD subfield X 100 <sub>16</sub> + Reference ROM data. Used in MAC microcode-assisted and CP copy state instruction operations to convert register designator into CST RAM entry address.																	
2	If Debug condition exists (CST 3.1) and Debug Lockout is inactive (CST 3.1), micrand sequence branches to 080 <sub>16</sub> + CD. Debug Lockout activates.																	
	If conditions are not met, CST RAM address increments.																	
3	If Debug condition exists and Debug Lockout is inactive, micrand sequence branches to 0C0 <sub>16</sub> + CD. Debug Lockout activates.																	
	If Debug condition exists and Debug Lockout is active, micrand sequence branches to 7FF <sub>16</sub> .																	
	If neither case applies, exit occurs.																	

Table 3-2. General Micrand Bits and Descriptions (Sheet 2 of 11)

Bit	Name	Description
		<p><u>Decode</u>                      <u>Description</u></p> <p>6    If Debug condition exists and Debug Lockout is inactive, micrand sequence jumps to <math>180_{16} + CD</math>. Debug Lockout activates.</p> <p>      If conditions are not met, CST RAM address increments.</p> <p>7    If Debug condition exists and Debug Lockout is inactive, micrand sequence branches to <math>1C0_{16} + CD</math>. Debug Lockout activates.</p> <p>      If Debug condition exists and Debug Lockout is active, micrand sequence branches to <math>7FF_{16}</math>.</p> <p>      If neither case applies, exit occurs.</p> <p>If <math>CB=1</math> and <math>CA \neq 0</math>, decoded CS subfield (CST 3.2) selects condition which must be met before BD value (CST 3.4) is added to current CST RAM address.</p> <p><u>Decode</u>                      <u>Description</u></p> <p>0    Processor Halted (CST 3.0)</p> <p>1    If C170 mode, ECS Halt Exit (MAC 3.7). If C180 mode, BDP Bit 32.</p> <p>2    If C170 mode, ECS Error Exit (MAC 3.7). If C180 mode, BDP Bit 33.</p> <p>3    If C170 mode, ECS Present (MAC 3.11). If C180 mode, BDP Bit 34.</p> <p>4    DAI Register Bit 60 (OPI 3.12).</p> <p>5    DAI Register Bit 61 (OPI 3.12).</p> <p>6    DAI Register Bit 62 (OPI 3.12).</p> <p>7    DAI Register Bit 63 (OPI 3.12).</p> <p>8    A Start &gt; A Terminate (OPI 3.14).</p> <p>9    X Start &gt; X Terminate (OPI 3.14).</p> <p>A    A Start &gt; zero and &lt; <math>F_{16}</math> (OPI 3.14).</p> <p>B    Untranslatable VMID (SM 3.0).</p> <p>C    Initial Ring = Final Ring (SM 3.2).</p> <p>D    Test Mode (MAC 3.11).</p>

Table 3-2. General Micrand Bits and Descriptions (Sheet 3 of 11)

Bit	Name	Description
		<p><u>Decode</u>                      <u>Description</u></p> <p>E     C170 Mode (SM 3.0).</p> <p>F     Inactive DAI Bit 33 (OPI 3.12).</p> <p>10    Inactive Processor Halted (CST 3.0).</p> <p>11    Inactive Condition 1.</p> <p>12    Inactive Condition 2.</p> <p>13    Inactive Condition 3.</p> <p>14    Inactive Condition 4.</p> <p>15    Inactive Condition 5.</p> <p>16    Inactive Condition 6.</p> <p>17    Inactive Condition 7.</p> <p>18    A Start not equal to A Terminate (OPI 3.14).</p> <p>19    X Start not equal to X Terminate (OPI 3.14).</p> <p>1A    Condition False.</p> <p>1B    C180 Monitor Mode.</p> <p>1C    IF Local Privilege Segment (SM 3.1).</p> <p>1D    IF Global Privilege Segment (SM 3.1).</p> <p>1E    MAC Busy (MAC 3.7).</p> <p>1F    LM Request outstanding (LM 3.8).</p> <p>If selected condition is not met, alternate action is based on CA subfield value (CST 3.1).</p> <p><u>Decode</u>                      <u>Description</u></p> <p>1     If condition is not met, current CST RAM address increments.</p> <p>2     If condition is not met, CST RAM address in Return Mux Register (CST 3.4) enters CST RAM.</p> <p>3     If condition is not met, exit occurs.</p> <p>Load Return Register (LRT) Field Decoder (CST 3.1) determines which value enters Return Mux Register.</p>

Table 3-2. General Micrand Bits and Descriptions (Sheet 4 of 11)

Bit	Name	Description																				
		<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>Inhibit load.</td></tr><tr><td>1</td><td>Load Return Mux Register with current address.</td></tr><tr><td>2</td><td>Inhibit load and set MAC Busy (MAC 3.7).</td></tr><tr><td>3</td><td>Load Return Mux Register with incremented CST RAM address.</td></tr></table> <p>MISC Field Decoder (CST 3.1) determines which of four miscellaneous occurrences will take place.</p> <table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>No operation.</td></tr><tr><td>1</td><td>Set Processor Halted FF (CST 3.0) if pipeline is empty and disable CST Rank 22 Valid (CST 3.3).</td></tr><tr><td>2</td><td>If Debug condition exists and Debug Lockout is inactive, unbranch and Register File (OPI 3.5) write operations associated with current micrand are suppressed.</td></tr><tr><td>3</td><td>Inhibit micrand retry.</td></tr></table>	<u>Decode</u>	<u>Description</u>	0	Inhibit load.	1	Load Return Mux Register with current address.	2	Inhibit load and set MAC Busy (MAC 3.7).	3	Load Return Mux Register with incremented CST RAM address.	<u>Decode</u>	<u>Description</u>	0	No operation.	1	Set Processor Halted FF (CST 3.0) if pipeline is empty and disable CST Rank 22 Valid (CST 3.3).	2	If Debug condition exists and Debug Lockout is inactive, unbranch and Register File (OPI 3.5) write operations associated with current micrand are suppressed.	3	Inhibit micrand retry.
<u>Decode</u>	<u>Description</u>																					
0	Inhibit load.																					
1	Load Return Mux Register with current address.																					
2	Inhibit load and set MAC Busy (MAC 3.7).																					
3	Load Return Mux Register with incremented CST RAM address.																					
<u>Decode</u>	<u>Description</u>																					
0	No operation.																					
1	Set Processor Halted FF (CST 3.0) if pipeline is empty and disable CST Rank 22 Valid (CST 3.3).																					
2	If Debug condition exists and Debug Lockout is inactive, unbranch and Register File (OPI 3.5) write operations associated with current micrand are suppressed.																					
3	Inhibit micrand retry.																					
16	Point of No Return (PONR)	Defines point of time during instruction execution when exceptions are evaluated for interrupt or retry operations (ICC 3.0-3.2).																				
17	Clock Condition Register (CCR)	Clocks exception indications into Monitor Condition Register (ICC 3.1) and User Condition Register (ICC 3.2). Signifies end of instruction. CCR enters ICC from Rank 50 Instruction Control Signal Register (ICP 3.3).																				
18-20	Counter Control	Used to manipulate OPI-resident counters (OPI 3.13, 3.14) during multiple register load/store operations and exchanges.																				
		<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>No change.</td></tr><tr><td>1</td><td>Set N Counter (OPI 3.14) to 21<sub>16</sub>, A Start Counter (OPI 3.14) to 16<sub>16</sub>, and X Start Counter (OPI 3.14) to C<sub>16</sub>.</td></tr><tr><td>2</td><td>Load X Start Counter, A Terminate Register (OPI 3.14) and X Terminate Register (OPI 3.14) per RDSc field. Clear N Counter.</td></tr></table>	<u>Decode</u>	<u>Description</u>	0	No change.	1	Set N Counter (OPI 3.14) to 21 <sub>16</sub> , A Start Counter (OPI 3.14) to 16 <sub>16</sub> , and X Start Counter (OPI 3.14) to C <sub>16</sub> .	2	Load X Start Counter, A Terminate Register (OPI 3.14) and X Terminate Register (OPI 3.14) per RDSc field. Clear N Counter.												
<u>Decode</u>	<u>Description</u>																					
0	No change.																					
1	Set N Counter (OPI 3.14) to 21 <sub>16</sub> , A Start Counter (OPI 3.14) to 16 <sub>16</sub> , and X Start Counter (OPI 3.14) to C <sub>16</sub> .																					
2	Load X Start Counter, A Terminate Register (OPI 3.14) and X Terminate Register (OPI 3.14) per RDSc field. Clear N Counter.																					

Table 3-2. General Micrand Bits and Descriptions (Sheet 5 of 11)

Bit	Name	Description																		
21-23	Functional Unit Field	<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>3</td><td>Load X Start Counter, A Terminate Register and X Terminate Register per RDSc. Clear A Start and N counters. Instruction Specification Error activates if A Terminate is less than 2.</td></tr><tr><td>4</td><td>Increment A Start and N counters.</td></tr><tr><td>5</td><td>Increment X Start and N counters.</td></tr><tr><td>6</td><td>Increment N Counter.</td></tr><tr><td>7</td><td>Load X Start Counter, A Terminate Register and X Terminate Register per RDSc. Clear A Start and N counters. Environment Specification Error activates if A Terminate is less than 2.</td></tr></table>	<u>Decode</u>	<u>Description</u>	3	Load X Start Counter, A Terminate Register and X Terminate Register per RDSc. Clear A Start and N counters. Instruction Specification Error activates if A Terminate is less than 2.	4	Increment A Start and N counters.	5	Increment X Start and N counters.	6	Increment N Counter.	7	Load X Start Counter, A Terminate Register and X Terminate Register per RDSc. Clear A Start and N counters. Environment Specification Error activates if A Terminate is less than 2.						
		<u>Decode</u>	<u>Description</u>																	
		3	Load X Start Counter, A Terminate Register and X Terminate Register per RDSc. Clear A Start and N counters. Instruction Specification Error activates if A Terminate is less than 2.																	
		4	Increment A Start and N counters.																	
		5	Increment X Start and N counters.																	
		6	Increment N Counter.																	
		7	Load X Start Counter, A Terminate Register and X Terminate Register per RDSc. Clear A Start and N counters. Environment Specification Error activates if A Terminate is less than 2.																	
		Determines which Functional Unit or units are to receive a Go signal and employ Functional Unit Micrand. Decode occurs in General Micrand Functional Unit Decoder (ICP 3.5) and causes appropriate Function Unit Go signal to be enabled when Write Data Select field (WDS) does not equal 0.																		
		<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>Issue to ALN.</td></tr><tr><td>1</td><td>Issue to ALN and load ALN Shift Count Register (AC 3.14).</td></tr><tr><td>2</td><td>Issue to BDP.</td></tr><tr><td>3</td><td>Issue to BDP if AC A and B streams and BDP are not busy (ICP 3.5).</td></tr><tr><td>4</td><td>Issue to AC B Stream if AC B Stream is not busy (ICP 3.5).</td></tr><tr><td>5</td><td>Issue to AC B Stream if AC A and B streams are not busy.</td></tr><tr><td>6</td><td>Issue to AC A Stream and LM if AC A and B streams and LM are not busy.</td></tr><tr><td>7</td><td>Issue to AC A Stream if AC A and B streams are not busy.</td></tr></table>	<u>Decode</u>	<u>Description</u>	0	Issue to ALN.	1	Issue to ALN and load ALN Shift Count Register (AC 3.14).	2	Issue to BDP.	3	Issue to BDP if AC A and B streams and BDP are not busy (ICP 3.5).	4	Issue to AC B Stream if AC B Stream is not busy (ICP 3.5).	5	Issue to AC B Stream if AC A and B streams are not busy.	6	Issue to AC A Stream and LM if AC A and B streams and LM are not busy.	7	Issue to AC A Stream if AC A and B streams are not busy.
		<u>Decode</u>	<u>Description</u>																	
0	Issue to ALN.																			
1	Issue to ALN and load ALN Shift Count Register (AC 3.14).																			
2	Issue to BDP.																			
3	Issue to BDP if AC A and B streams and BDP are not busy (ICP 3.5).																			
4	Issue to AC B Stream if AC B Stream is not busy (ICP 3.5).																			
5	Issue to AC B Stream if AC A and B streams are not busy.																			
6	Issue to AC A Stream and LM if AC A and B streams and LM are not busy.																			
7	Issue to AC A Stream if AC A and B streams are not busy.																			
When WDS field equals 0, Functional Unit Field decodes of 0, 1, 4-7 are illegal.																				
<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>2</td><td>No functional unit is selected. No Execution Response (ICP 3.6) activates.</td></tr><tr><td>3</td><td>No functional unit is selected. Internal Response (ICP 3.6) is generated when BDP is not busy.</td></tr></table>	<u>Decode</u>	<u>Description</u>	2	No functional unit is selected. No Execution Response (ICP 3.6) activates.	3	No functional unit is selected. Internal Response (ICP 3.6) is generated when BDP is not busy.														
<u>Decode</u>	<u>Description</u>																			
2	No functional unit is selected. No Execution Response (ICP 3.6) activates.																			
3	No functional unit is selected. Internal Response (ICP 3.6) is generated when BDP is not busy.																			

Table 3-2. General Micrand Bits and Descriptions (Sheet 6 of 11)

Bit	Name	Description
24, 25	WDS Field	Controls selection of data to be entered into DAI Register (OPI 3.12).  <div> <div>Decode</div> <div>Description</div> </div> <div> <div>0</div> <div>Functional Unit Micrand is selected for entry into Register File location specified by RDSa. Functional Unit Go is inhibited.</div> </div> <div> <div>1</div> <div>ALN Result data.</div> </div> <div> <div>2</div> <div>Load data from AC.</div> </div> <div> <div>3</div> <div>LM Read data.</div> </div>
26-28	Immediate Data Select (IDS)	Selects immediate operand to be used as AC address offset.  <div> <div>Decode</div> <div>Description</div> </div> <div> <div>0</div> <div>C180 D</div> </div> <div> <div>1</div> <div>C180 Q, BDP Offset</div> </div> <div> <div>2</div> <div>Zero</div> </div> <div> <div>3</div> <div>Literal 8</div> </div> <div> <div>4</div> <div>C180 D X 8</div> </div> <div> <div>5</div> <div>C180 Q X 8</div> </div> <div> <div>6</div> <div>C180 B Operand X 8</div> </div> <div> <div>7</div> <div>C170 K X 8</div> </div>
29-39 41-51 53-61	Register/Data Select (RDS) a, b, c	Used to perform Live Register (OPI 3.18) and Register File (OPI 3.5) read operations. Fields also provide literal data and immediate operands directly to functional units. Subfields select Live Register data, immediate operand data, or address of Register File location to be read. RDSa also may be configured to perform write operations.  Subfields that control read operations include Register Data (RD), Literal Register File Address (LT), Literal/Other Data (LO), Data/Live Selector (DL), Data Shift (DS), Data Subfunction Select (DSS), Live Register Read (LRR), Register Addressing Function (RAF), and Unless Zero (UZ). Bias (BI), Literal Register Address (LRA), Literal Data (LD), and C170 Register Indicator (OR) subfields may be used in Register File address calculations.

Table 3-2. General Micrand Bits and Descriptions (Sheet 7 of 11)

Bit	Name	Description																												
		<p>If RD=0 and LT=0, combination of BI, OR, and RAF determine Register File address in Register File Address Selector (OPI 3.3). BI provides two most significant bits of address. OR determines next most significant bit. If set, OR indicates C170 A or X register; if clear, it indicates C170 B Register or C180 address. RAF selects register designator from instruction or appropriate counter as low-order four bits of address.</p> <table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>j</td></tr><tr><td>1</td><td>k</td></tr><tr><td>2</td><td>C170 i</td></tr><tr><td>3</td><td>A Start Counter (OPI 3.14)</td></tr><tr><td>4</td><td>C180 j+1</td></tr><tr><td>5</td><td>C180 k+1</td></tr><tr><td>6</td><td>C180 i</td></tr><tr><td>7</td><td>X Start Counter (OPI 3.14)</td></tr></table> <p>Active UZ forces Register File output to zero and disables conflict checks (OPI 3.2).</p> <p>If RD=0 and LT=1, combination of BI and LRA provides literal values which become Register File read address. UZ performs operation described in previous format.</p> <p>If RD=1 and LO=0, LD becomes Immediate Operand Register Bits 57-63 (OPI 3.7). Remaining 57 bits are forced to zero. Typically, LD is mask or constant for arithmetic operations in ALN or it indicates field length to AC.</p> <p>If RD=1, LO=1, and DL=0, DSS subfield selects designator from instruction or contents of N Register for output to functional unit. Selection occurs in Immediate Operand multiplexers (OPI 3.6).</p> <table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>Not used</td></tr><tr><td>1</td><td>C170 RAC X 8</td></tr><tr><td>2</td><td>C180 D</td></tr><tr><td>3</td><td>C170 K</td></tr></table>	<u>Decode</u>	<u>Description</u>	0	j	1	k	2	C170 i	3	A Start Counter (OPI 3.14)	4	C180 j+1	5	C180 k+1	6	C180 i	7	X Start Counter (OPI 3.14)	<u>Decode</u>	<u>Description</u>	0	Not used	1	C170 RAC X 8	2	C180 D	3	C170 K
<u>Decode</u>	<u>Description</u>																													
0	j																													
1	k																													
2	C170 i																													
3	A Start Counter (OPI 3.14)																													
4	C180 j+1																													
5	C180 k+1																													
6	C180 i																													
7	X Start Counter (OPI 3.14)																													
<u>Decode</u>	<u>Description</u>																													
0	Not used																													
1	C170 RAC X 8																													
2	C180 D																													
3	C170 K																													

Table 3-2. General Micrand Bits and Descriptions (Sheet 8 of 11)

Bit	Name	Description																																										
		<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>4</td><td>C170 jk</td></tr><tr><td>5</td><td>C180 jk, BDP Length</td></tr><tr><td>6</td><td>C180 j</td></tr><tr><td>7</td><td>N Register (OPI 3.14)</td></tr><tr><td>8</td><td>Rank 22 P</td></tr><tr><td>9</td><td>C180 jkQ</td></tr><tr><td>A</td><td>C180 Q</td></tr></table> <p>If RD=1, LO=1, and DL=1, LRR subfield selects Live Register read data in Live Register Read Control (OPI 3.18).</p> <table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>Rank 50 P (ICP 3.0) and Old P (SM 3.1).</td></tr><tr><td>4</td><td>Rank 50 UTP (ICP 3.0) (restricted to selection by RDSb and RDSc).</td></tr><tr><td>8</td><td>PIT.</td></tr><tr><td>9</td><td>SIT.</td></tr><tr><td>A</td><td>UCR.</td></tr><tr><td>B</td><td>MCR.</td></tr><tr><td>C</td><td>Exchange Address and BDP X1 Result.</td></tr><tr><td>D</td><td>BDP X0 and X1 results and Exchange Code.</td></tr><tr><td>E</td><td>Old P Ring number and BDP X1 Result.</td></tr><tr><td>F</td><td>Exit Mode Condition and BDP X1 Result.</td></tr><tr><td>1A</td><td>UCR (and clear UCR) (selection restricted to RDSb and RDSc).</td></tr><tr><td>2B</td><td>MCR (and clear MCR) (restricted to selection by RDSb and RDSc).</td></tr></table> <p>DS field applies to all five read formats. It is operational for RDSa and RDSb only. DS controls left shifting of Operand Data in Operand Left Shift Network (OPI 3.10).</p>	<u>Decode</u>	<u>Description</u>	4	C170 jk	5	C180 jk, BDP Length	6	C180 j	7	N Register (OPI 3.14)	8	Rank 22 P	9	C180 jkQ	A	C180 Q	<u>Decode</u>	<u>Description</u>	0	Rank 50 P (ICP 3.0) and Old P (SM 3.1).	4	Rank 50 UTP (ICP 3.0) (restricted to selection by RDSb and RDSc).	8	PIT.	9	SIT.	A	UCR.	B	MCR.	C	Exchange Address and BDP X1 Result.	D	BDP X0 and X1 results and Exchange Code.	E	Old P Ring number and BDP X1 Result.	F	Exit Mode Condition and BDP X1 Result.	1A	UCR (and clear UCR) (selection restricted to RDSb and RDSc).	2B	MCR (and clear MCR) (restricted to selection by RDSb and RDSc).
<u>Decode</u>	<u>Description</u>																																											
4	C170 jk																																											
5	C180 jk, BDP Length																																											
6	C180 j																																											
7	N Register (OPI 3.14)																																											
8	Rank 22 P																																											
9	C180 jkQ																																											
A	C180 Q																																											
<u>Decode</u>	<u>Description</u>																																											
0	Rank 50 P (ICP 3.0) and Old P (SM 3.1).																																											
4	Rank 50 UTP (ICP 3.0) (restricted to selection by RDSb and RDSc).																																											
8	PIT.																																											
9	SIT.																																											
A	UCR.																																											
B	MCR.																																											
C	Exchange Address and BDP X1 Result.																																											
D	BDP X0 and X1 results and Exchange Code.																																											
E	Old P Ring number and BDP X1 Result.																																											
F	Exit Mode Condition and BDP X1 Result.																																											
1A	UCR (and clear UCR) (selection restricted to RDSb and RDSc).																																											
2B	MCR (and clear MCR) (restricted to selection by RDSb and RDSc).																																											



Table 3-2. General Micrand Bits and Descriptions (Sheet 9 of 11)

Bit	Name	Description																		
29-39 AC MICRAND 53-63	Register/Data Select (RDS) a,d	<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>No shift.</td></tr><tr><td>1</td><td>Left shift 1 of low-order 32 bits.</td></tr><tr><td>2</td><td>Left shift 3 of low-order 32 bits.</td></tr><tr><td>3</td><td>Left shift 3 of low-order 18 bits.</td></tr></table>	<u>Decode</u>	<u>Description</u>	0	No shift.	1	Left shift 1 of low-order 32 bits.	2	Left shift 3 of low-order 32 bits.	3	Left shift 3 of low-order 18 bits.								
		<u>Decode</u>	<u>Description</u>																	
0	No shift.																			
1	Left shift 1 of low-order 32 bits.																			
2	Left shift 3 of low-order 32 bits.																			
3	Left shift 3 of low-order 18 bits.																			
		Used to perform Live Register and Register File write operations. Subfields determine Live Register or Register File location to be written. RDSa also may be configured to perform read operations.																		
		Subfields that control write operations are Write Width into Register File (WW), Live Register Write/Complete Cycle Control (LRW/CSC), RAF, RD, and LT. BI, OR, and LRA subfields may be used in Register File write address calculations. Definitions of RAF, RD, LT, BI, OR, and LRA are same as for RDSa, b, c read operations.																		
		If RD=0, RSDa or RSDd designates which Register File location is to be written. BI, OR, and RAF or BI and LRA pass through Minipipe (OPI 3.1) to address Register File in writing phase of micrand execution.																		
		WW determines which bits in a 64-bit Register File location are to be written. WW field decode occurs in Register File Write Width Control (OPI 3.4).																		
		<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>Not used.</td></tr><tr><td>1</td><td>48 low-order bits. Select P ring number or data ring number, whichever is larger, in Largest Ring Selector (OPI 3.12).</td></tr><tr><td>2</td><td>32 low-order bits.</td></tr><tr><td>3</td><td>48 low-order bits. Select Rank 41 Largest Ring or data ring, whichever is larger, in Largest Ring Selector. Ring=0 test is enabled.</td></tr><tr><td>4</td><td>Byte is selected by MAC. Refer to Register File Write Control, OPI 3.4.</td></tr><tr><td>5</td><td>48 low-order bits.</td></tr><tr><td>6</td><td>16 high-order bits.</td></tr><tr><td>7</td><td>64 bits.</td></tr></table>	<u>Decode</u>	<u>Description</u>	0	Not used.	1	48 low-order bits. Select P ring number or data ring number, whichever is larger, in Largest Ring Selector (OPI 3.12).	2	32 low-order bits.	3	48 low-order bits. Select Rank 41 Largest Ring or data ring, whichever is larger, in Largest Ring Selector. Ring=0 test is enabled.	4	Byte is selected by MAC. Refer to Register File Write Control, OPI 3.4.	5	48 low-order bits.	6	16 high-order bits.	7	64 bits.
<u>Decode</u>	<u>Description</u>																			
0	Not used.																			
1	48 low-order bits. Select P ring number or data ring number, whichever is larger, in Largest Ring Selector (OPI 3.12).																			
2	32 low-order bits.																			
3	48 low-order bits. Select Rank 41 Largest Ring or data ring, whichever is larger, in Largest Ring Selector. Ring=0 test is enabled.																			
4	Byte is selected by MAC. Refer to Register File Write Control, OPI 3.4.																			
5	48 low-order bits.																			
6	16 high-order bits.																			
7	64 bits.																			

Table 3-2. General Micrand Bits and Descriptions (Sheet 10 of 11)

Bit	Name	Description																																																		
		<p>If RD=1, LRW/CSC decode determines which Live Register write or micrand completion control activity will occur. Decoding occurs in Live Register Write Decoder, OPI 3.15, unless otherwise indicated.</p> <table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>C</td><td>Load STA upper.</td></tr><tr><td>D</td><td>Load STA lower.</td></tr><tr><td>E</td><td>Load Debug Mask Register.</td></tr><tr><td>11</td><td>Load Trap Mask Register.</td></tr><tr><td>12</td><td>Load UMR.</td></tr><tr><td>13</td><td>Load MMR.</td></tr><tr><td>14</td><td>Load UCR.</td></tr><tr><td>15</td><td>Load MCR.</td></tr><tr><td>1A</td><td>Load PIT upper.</td></tr><tr><td>1B</td><td>Load PIT lower.</td></tr><tr><td>1F</td><td>Load STL Register.</td></tr><tr><td>28-2F</td><td>MAC operation.</td></tr><tr><td>30</td><td>Load C170 RAC Register.</td></tr><tr><td>31</td><td>Gate RAC + FLC.</td></tr><tr><td>32</td><td>Load EMMR.</td></tr><tr><td>34</td><td>Purge instruction stack.</td></tr><tr><td>35</td><td>Load SIT.</td></tr><tr><td>36</td><td>Load C170 Monitor Mode.</td></tr><tr><td>38</td><td>Load Reference ROM Address Register.</td></tr><tr><td>39</td><td>Load PSM Register.</td></tr><tr><td>3A</td><td>Load DAI Condition Register.</td></tr><tr><td>3B</td><td>Enable toggle of C180 Monitor/Job Mode.</td></tr><tr><td>3C</td><td>Set Exchange Mode.</td></tr><tr><td>3D</td><td>IOU Exchange Accept.</td></tr></table>	<u>Decode</u>	<u>Description</u>	C	Load STA upper.	D	Load STA lower.	E	Load Debug Mask Register.	11	Load Trap Mask Register.	12	Load UMR.	13	Load MMR.	14	Load UCR.	15	Load MCR.	1A	Load PIT upper.	1B	Load PIT lower.	1F	Load STL Register.	28-2F	MAC operation.	30	Load C170 RAC Register.	31	Gate RAC + FLC.	32	Load EMMR.	34	Purge instruction stack.	35	Load SIT.	36	Load C170 Monitor Mode.	38	Load Reference ROM Address Register.	39	Load PSM Register.	3A	Load DAI Condition Register.	3B	Enable toggle of C180 Monitor/Job Mode.	3C	Set Exchange Mode.	3D	IOU Exchange Accept.
<u>Decode</u>	<u>Description</u>																																																			
C	Load STA upper.																																																			
D	Load STA lower.																																																			
E	Load Debug Mask Register.																																																			
11	Load Trap Mask Register.																																																			
12	Load UMR.																																																			
13	Load MMR.																																																			
14	Load UCR.																																																			
15	Load MCR.																																																			
1A	Load PIT upper.																																																			
1B	Load PIT lower.																																																			
1F	Load STL Register.																																																			
28-2F	MAC operation.																																																			
30	Load C170 RAC Register.																																																			
31	Gate RAC + FLC.																																																			
32	Load EMMR.																																																			
34	Purge instruction stack.																																																			
35	Load SIT.																																																			
36	Load C170 Monitor Mode.																																																			
38	Load Reference ROM Address Register.																																																			
39	Load PSM Register.																																																			
3A	Load DAI Condition Register.																																																			
3B	Enable toggle of C180 Monitor/Job Mode.																																																			
3C	Set Exchange Mode.																																																			
3D	IOU Exchange Accept.																																																			

Table 3-2. General Micrand Bits and Descriptions (Sheet 11 of 11)

Bit	Name	Description
40,52 AC MICRAND	RDSb, c, d Sign Extension	<p><u>Decode</u>                      <u>Description</u></p> <p>80    Set Stream Mode (OPI 3.16).</p> <p>100   ICP Branch Index Request (ICP 3.3).</p> <p>200   Clear Stream Mode (OPI 3.16).</p> <p>Enables sign extension of C170 60-bit words to 64 bits. Bit 40 also specifies AC Address Adder (AC 1.0) subtract operation.</p>
		<p>62, 63    General Micrand Extension</p> <p>Expands General Micrand control to include additional miscellaneous operations. Decoding occurs in General Micrand Extension Decoder (ICP 3.6).</p> <p><u>Decode</u>                      <u>Description</u></p> <p>0    No operation.</p> <p>1    If Functional Unit field (General Micrand Bits 21-23) equals 0 or 1, decode is illegal.</p> <p>If Functional Unit field equals 2 or 3, decode disables BDP Go (ICP 3.5) and generates Wait Second Response (ICP 3.6).</p> <p>If Functional Unit field equals 4-7, decode generates Hold UTP Segment Flag (ICP 3.0), which becomes Rank 50 UTP Bit 29.</p> <p><u>Decode</u>                      <u>Description</u></p> <p>2    Suspends instruction fetch operations (IF 3.6) and clears instruction control pipeline beginning with rank 13 (ICP 3.4). Instruction fetching resumes when addressing of Live Register write address 100<sub>16</sub> permits new P to enter IF.</p> <p>3    If Functional Unit field equals 0, 2-5, operation is undefined.</p> <p>If Functional Unit field equals 1, shift count enters ALN Shift Count Register (AC 3.14).</p> <p>If Functional Unit field equals 6 or 7, Multiple Response Flag indicates micrand operation requires two responses (ICP 3.5).</p>

## AC/LM/SM Micrand

This 64-bit Functional Unit Micrand provides major cycle control for AC (bytes 0, 1), LM (bytes 2, 3), and SM (bytes 4, 5). Bytes 6 and 7 extend the General Micrand by providing control for loading the Register File and Live Registers in OPI with memory data. These bytes are described in the previous paragraph (General Micrand). A General Micrand FU field value of four or more enables the AC/LM/SM micrand. The AC/LM/SM micrand (bytes 0 through 5) is shown in figure 3-17 and described in table 3-3.

Table 3-3. AC/LM/SM Micrand Bits and Descriptions (Sheet 1 of 6)

Bit	Name	Description										
0-3	Partial Soft Control Pointer	Partially points to the first soft control address of an AC sequence (AC 3.1, 2, 3).										
4	Memory Write	Enables writing data into LM/CM (AC 3.4, LM 3.2).										
5	Address Restriction	Enables setting Address Restriction FF (AC 3.18) when Address Adder (AC 3.14) bits 29-31 are clear (Micrand Bit 8 clear) or bit 31 is clear (Micrand Bit 8 set).										
6	Byte Assembly/Disassembly	Enables disassembly of words from LM (A, B streams) and assembly of bytes from BDP (C stream) (AC 3.2, 7, 13, 15, 16).										
7	Left to Right Addressing	Enables left-to-right (set) or right-to-left (clear) byte addressing in memory (LM 3.0, AC 3.5, 11, 15).										
8	Address Modulus Designator	When set, enables setting Address Restriction FF (AC 3.18) when Address Adder (AC 3.14) Bit 31 is clear. When clear, enables setting Address Restriction FF when Address Adder bits 29-31 are clear.										
9	Address Length	Enables stream length to A input of Address Adder (AC 3.4, 14).										
10	Address Specification Test	Generates Operand Address Specification Error if leftmost byte number address bit is set (AC 3.18).										
11	Page Valid	Causes LM to disregard valid bit when searching system page table (AC 3.16).										
12-14	Function Code	Expands micrand to enable various AC operations using Function Decoder (AC 3.0).										
		<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>Not used.</td></tr><tr><td>1</td><td>Table load.</td></tr><tr><td>2</td><td>Ak descriptor only.</td></tr><tr><td>3</td><td>Fast A stream.</td></tr></table>	<u>Decode</u>	<u>Description</u>	0	Not used.	1	Table load.	2	Ak descriptor only.	3	Fast A stream.
<u>Decode</u>	<u>Description</u>											
0	Not used.											
1	Table load.											
2	Ak descriptor only.											
3	Fast A stream.											

Table 3-3. AC/LM/SM Micrand Bits and Descriptions (Sheet 2 of 6)

Bit	Name	Description																												
		<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>4</td><td>Test and set bit.</td></tr><tr><td>5</td><td>BDP prevalidate store.</td></tr><tr><td>6</td><td>BDP multiply/divide not last word.</td></tr><tr><td>7</td><td>BDP multiply/divide not word.</td></tr></table>	<u>Decode</u>	<u>Description</u>	4	Test and set bit.	5	BDP prevalidate store.	6	BDP multiply/divide not last word.	7	BDP multiply/divide not word.																		
<u>Decode</u>	<u>Description</u>																													
4	Test and set bit.																													
5	BDP prevalidate store.																													
6	BDP multiply/divide not last word.																													
7	BDP multiply/divide not word.																													
15	Address Out of Range	Enables C170 Mode Range Tester (AC 3.15).																												
16,17		Not used.																												
18	Purge Cache	Enables purging one block (four 64-bit words) from Cache Memory (LM 3.2).																												
19		Not used.																												
20	Stream Mode	Enables Stream Mode Tag as address to write Register File (LM 3.7).																												
21-24	CMC Function Code	Sent to CMC to select CM function (LM 3.7).																												
		<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>Read.</td></tr><tr><td>1</td><td>Not used.</td></tr><tr><td>2</td><td>Write.</td></tr><tr><td>3</td><td>Not used.</td></tr><tr><td>4</td><td>Read and set lock.</td></tr><tr><td>5</td><td>Read and clear lock.</td></tr><tr><td>6</td><td>Exchange.</td></tr><tr><td>7</td><td>Not used.</td></tr><tr><td>8</td><td>Not used.</td></tr><tr><td>9</td><td>Not used.</td></tr><tr><td>A</td><td>Read Free Running Counter.</td></tr><tr><td>B</td><td>Resynchronize Refresh Counter.</td></tr><tr><td>C</td><td>Interrupt.</td></tr></table>	<u>Decode</u>	<u>Description</u>	0	Read.	1	Not used.	2	Write.	3	Not used.	4	Read and set lock.	5	Read and clear lock.	6	Exchange.	7	Not used.	8	Not used.	9	Not used.	A	Read Free Running Counter.	B	Resynchronize Refresh Counter.	C	Interrupt.
<u>Decode</u>	<u>Description</u>																													
0	Read.																													
1	Not used.																													
2	Write.																													
3	Not used.																													
4	Read and set lock.																													
5	Read and clear lock.																													
6	Exchange.																													
7	Not used.																													
8	Not used.																													
9	Not used.																													
A	Read Free Running Counter.																													
B	Resynchronize Refresh Counter.																													
C	Interrupt.																													

Table 3-3. AC/LM/SM Micrand Bits and Descriptions (Sheet 3 of 6)

Bit	Name	Description										
		<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>D</td><td>Not used.</td></tr><tr><td>E</td><td>Not used.</td></tr><tr><td>F</td><td>Not used.</td></tr></table>	<u>Decode</u>	<u>Description</u>	D	Not used.	E	Not used.	F	Not used.		
<u>Decode</u>	<u>Description</u>											
D	Not used.											
E	Not used.											
F	Not used.											
25	Return Page Table Index	If real memory address (RMA) operation, disables CMC Mark Bits 0 and 1 (LM 3.7). If normal memory operation, enables setting Page Table Hit FF (LM 3.11).										
26	Translate System Virtual Address (SVA) to RMA	Generates Test signal (LM 3.2).										
27	Suppress Page Table Search Without Find	Disables LM Page Table Search Without Find signal to ICC (LM 3.13).										
28	Response Director	Directs response to AC (set) or ICP (clear) during memory read operations (LM 3.0,4).										
29	Initialize Page Table	Enables loading of Page Table Address and Length registers (LM 3.10).										
30, 31	Cache Read	Enables cache read operation (both bits set) (AC 3.4).										
32	Purge Per Code	Enables Purge Code from ICP (SM 3.4).										
33, 34	Explicit Purge Code	Purges Cache Memory, Segment Maps, and Page Map (SM 3.4). <table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>None.</td></tr><tr><td>1</td><td>Purge all of Cache Memory.</td></tr><tr><td>2</td><td>Purge all of Segment Map.</td></tr><tr><td>3</td><td>Purge all of Segment and Page Maps.</td></tr></table>	<u>Decode</u>	<u>Description</u>	0	None.	1	Purge all of Cache Memory.	2	Purge all of Segment Map.	3	Purge all of Segment and Page Maps.
<u>Decode</u>	<u>Description</u>											
0	None.											
1	Purge all of Cache Memory.											
2	Purge all of Segment Map.											
3	Purge all of Segment and Page Maps.											
35	RMA	Defines associated address as an RMA which disables virtual address conversion (SM 3.4, LM 3.1).										
36	SVA	Defines associated A Register address as an SVA rather than a process virtual address (PVA) (SM 3.4).										
37	Invalid Segment	Enables Invalid Segment Test (SM 3.4).										
38	Ring and Segment Selector	Selects ring and segment number from P Register (set) or Register File (clear) (SM 3.4).										

Table 3-3. AC/LM/SM Micrand Bits and Descriptions (Sheet 4 of 6)

Bit	Name	Description
39	Allocate	Enables allocation of SM (SM 3.3, 4).
40	Read Validate	Indicates operation requires read segment validation (SM 3.5).
41	Write Validate	Indicates operation requires write segment validation (SM 3.5).
42		Not used.
43-46	Function Code	Determines SM operation using Micrand Decoder (SM 3.4).
<u>Bit 46 Clear</u>		
<u>Decode</u>		<u>Description</u>
0		Not used.
1		Call 1 (Call Binding Section Validation) - Loads largest ring (P or PVA) into Ring Hold Register (SM 3.2). Test for binding section segment (read privilege equals three) (SM 3.5).
2		Call 2 (Code Base Pointer Processing) - Test for P ring less than R1 on outward call (SM 3.5). Test for nonexecutable segment (execute privilege equals zero) (SM 3.5). Test and load new P key/lock (SM 3.1). Test for invalid segment (ring equals zero) (SM 3.5). Load New P and P Descriptor Save Registers (SM 3.1).
3		Pop - Test for inter-ring pop (old P ring not equal to PVA ring) (SM 3.2).
4		Intersegment Branch - Select P ring to New P Register (SM 3.2). Test and load new P key/lock (SM 3.1). Test for ring less than R1 or greater than R2 (SM 3.5). Test for nonexecutable segment (execute privilege equals zero) (SM 3.5). Load New P and P Descriptor Save Registers (SM 3.1).
5		Return 2 (Third Return Micrand) - Select R2 ring to New P Register (SM 3.2). Test and load new P key/lock (SM 3.1). Test for ring less than R1 or greater than R2 (SM 3.5). Test for hold ring greater than PVA ring on inward return (SM 3.2). Test for nonexecutable segment (execute privilege equals zero) (SM 3.5). Load New P and P Descriptor Save Registers (SM 3.1).
6		New P to Old P - Load Old P Register (SM 3.1).

Table 3-3. AC/LM/SM Micrand Bits and Descriptions (Sheet 5 of 6)

Bit	Name	Description
		<p><u>Decode</u>                      <u>Description</u></p> <p>7    Exchange (Second Exchange Micrand) - Select PVA ring to New P Register (SM 3.2). Test for ring less than R1 or greater than R2 (SM 3.5). Test and load new P key/lock (SM 3.1). Test for nonexecutable segment (execute privilege equals zero) (SM 3.5). Load New P and P Descriptor Save Registers (SM 3.1).</p> <p><u>Bit 46 Set</u></p> <p><u>Decode</u>                      <u>Description</u></p> <p>0    Return 1 (First Return Micrand) - Loads PVA ring into Ring Hold Register (SM 3.2).</p> <p>1    Test Virtual Machine Identifier (VMID) (First Exchange and Second Return Micrands) - Loads Code Base Pointer VMID into Holding Register (SM 3.0).</p> <p>2    Force Allocate - Test Monitor/Job Status RAMs (SM 3.3) and allocate SM accordingly. Test Enable Segment Descriptor Mux parity errors (SM 3.5).</p> <p>3    Return Hit - Send SM Inter-ring Pop to ICP if valid SM hit occurs (SM 3.4).</p> <p>4    Force Write Job RAM - Enable writing test data into Job Status RAM (SM 3.3).</p> <p>5    Force Write Monitor RAM - Enable writing test data into Monitor Status RAM (SM 3.3).</p> <p style="text-align: center;">NOTE</p> <p style="padding-left: 80px;">Decodes 2-5 are used by diagnostics micro-code only.</p> <p>6    Enable C170 Arithmetic - Enables AC to perform C170 arithmetic (SM 3.4).</p> <p>7    Disable C170 Arithmetic - Enables AC to perform C180 arithmetic (SM 3.4).</p>
47	Cache Bypass	Enables LM to bypass Cache Memory on read/write operation (SM 3.5).



Table 3-3. AC/LM/SM Micrand Bits and Descriptions (Sheet 6 of 6)

Bit	Name	Description										
48, 49		Not used.										
50, 51		Zero Fill RDSc Data - Forces RDSc data fields from Register File (OPI 3.5) to zero (OPI 3.9).										
		<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>Inhibit force zero operation.</td></tr><tr><td>1</td><td>Force bits 32-60 to zero.</td></tr><tr><td>2</td><td>Force bits 0-15 to zero.</td></tr><tr><td>3</td><td>Not used.</td></tr></table>	<u>Decode</u>	<u>Description</u>	0	Inhibit force zero operation.	1	Force bits 32-60 to zero.	2	Force bits 0-15 to zero.	3	Not used.
<u>Decode</u>	<u>Description</u>											
0	Inhibit force zero operation.											
1	Force bits 32-60 to zero.											
2	Force bits 0-15 to zero.											
3	Not used.											

BDP Micrand

This 64-bit Functional Unit Micrand provides major cycle control for BDP. A General Micrand FU field value of 2 or 3 enables this micrand to execute BDP operations. The BDP micrand is shown in figure 3-17 and described in table 3-4.

Table 3-4. BDP Micrand Bits and Descriptions (Sheet 1 of 8)

Bit	Name	Description																		
0-2	Error Test Group	Forms most significant address bits for Instruction Specification Error RAM (BDP 3.2, 3). Specifies which error group an instruction belongs to in order to test for data type and field length violations.																		
		<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>Decimal sum, difference, product, quotient.</td></tr><tr><td>1</td><td>Decimal scale, scale rounded.</td></tr><tr><td>2</td><td>Numeric move.</td></tr><tr><td>3</td><td>Edit.</td></tr><tr><td>4</td><td>Immediate (j=0).</td></tr><tr><td>5</td><td>Immediate (j=1).</td></tr><tr><td>6</td><td>All other instructions.</td></tr><tr><td>7</td><td>Calculate subscript.</td></tr></table>	<u>Decode</u>	<u>Description</u>	0	Decimal sum, difference, product, quotient.	1	Decimal scale, scale rounded.	2	Numeric move.	3	Edit.	4	Immediate (j=0).	5	Immediate (j=1).	6	All other instructions.	7	Calculate subscript.
<u>Decode</u>	<u>Description</u>																			
0	Decimal sum, difference, product, quotient.																			
1	Decimal scale, scale rounded.																			
2	Numeric move.																			
3	Edit.																			
4	Immediate (j=0).																			
5	Immediate (j=1).																			
6	All other instructions.																			
7	Calculate subscript.																			

Table 3-4. BDP Micrand Bits and Descriptions (Sheet 2 of 8)

Bit	Name	Description														
3-6	Force Data Type	Replaces Aj and Ak Descriptor Types when selected by Micrand Bits 7-9 (BDP 3.4).														
7	Force Aj Type Input	Forces Micrand Bits 3-6 to replace Aj Descriptor Type input (BDP 3.4).														
8	Force Ak Type Input	Forces Micrand Bits 3-6 to replace Ak Descriptor Type input (BDP 3.4).														
9	Force Ak Type Output	Forces Micrand Bits 3-6 to replace Ak Descriptor Type output (BDP 3.4).														
10	Load Most Significant Non-zero Byte	Loads Most Significant Nonzero Byte Address into Buffer RAM Address Counter after Buffer RAM Input has completed (BDP 3.14).														
11	Reset Buffer RAM Address Counter	Resets Buffer Ram Address Counter after Buffer RAM input has completed (BDP 3.14).														
12	Fill Byte	Determines fill byte when A or B stream source field is exhausted (BDP 3.7,10). Fill byte is 00 when clear, 20 (hexadecimal) when set.														
13	Gate A Stream	Gates A Stream Stage 3 Data to Stage 4 (BDP 3.12).														
14	Gate B Stream	Gates B Stream Stage 3 Data to Stage 4 (BDP 3.12).														
15	Decimal ALU	Determines whether Decimal/Binary ALU will operate in decimal (set) or binary (clear) mode (BDP 3.12).														
16	Lower Digit Subtract	Produces Complement Lower signal (BDP 3.21) which determines whether Decimal/Binary ALU (BDP 3.12) operates in subtract (set) or add (clear) mode.														
17-19	Input Terminate	Controls what conditions cause A and B stream operations to terminate (BDP 3.32).														
		<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>Not used.</td></tr><tr><td>1</td><td>Aj source field exhausted.</td></tr><tr><td>2</td><td>Ak source field exhausted.</td></tr><tr><td>3</td><td>Aj and Ak source fields exhausted.</td></tr><tr><td>4</td><td>Ak destination field full.</td></tr><tr><td>5</td><td>Ak destination field full and Aj source field exhausted.</td></tr></table>	<u>Decode</u>	<u>Description</u>	0	Not used.	1	Aj source field exhausted.	2	Ak source field exhausted.	3	Aj and Ak source fields exhausted.	4	Ak destination field full.	5	Ak destination field full and Aj source field exhausted.
<u>Decode</u>	<u>Description</u>															
0	Not used.															
1	Aj source field exhausted.															
2	Ak source field exhausted.															
3	Aj and Ak source fields exhausted.															
4	Ak destination field full.															
5	Ak destination field full and Aj source field exhausted.															

Table 3-4. BDP Micrand Bits and Descriptions (Sheet 3 of 8)

Bit	Name	Description																							
20, 21	Condition Select	<table><tr><th>Decode</th><th>Description</th></tr><tr><td>6</td><td>Ak destination field full and Ak source field exhausted.</td></tr><tr><td>7</td><td>Ak destination field full and Aj and Ak source fields exhausted.</td></tr></table>	Decode	Description	6	Ak destination field full and Ak source field exhausted.	7	Ak destination field full and Aj and Ak source fields exhausted.																	
		Decode	Description																						
		6	Ak destination field full and Ak source field exhausted.																						
		7	Ak destination field full and Aj and Ak source fields exhausted.																						
		Selects status inputs to BDP Condition Bit Select Mux (BDP 3.14).																							
		<table><tr><th>Decode</th><th>Description</th></tr><tr><td></td><td><table><tr><th>Bit 1</th><th>Bit 2</th><th>Bit 3</th></tr><tr><td>0</td><td>Logical Zero</td><td>BDP Busy</td><td>Input Complete</td></tr><tr><td>1</td><td>Aj Source Decimal</td><td>Ak Source Decimal</td><td>BDP Busy</td></tr><tr><td>2</td><td>Abort Divide</td><td>Aj and Ak Unequal</td><td>More Than 9 Bytes in Buffer</td></tr><tr><td>3</td><td>Aj Source Decimal</td><td>Aj Source Sign Negative</td><td>More Than 9 Bytes in Buffer</td></tr></table></td></tr></table>	Decode	Description		<table><tr><th>Bit 1</th><th>Bit 2</th><th>Bit 3</th></tr><tr><td>0</td><td>Logical Zero</td><td>BDP Busy</td><td>Input Complete</td></tr><tr><td>1</td><td>Aj Source Decimal</td><td>Ak Source Decimal</td><td>BDP Busy</td></tr><tr><td>2</td><td>Abort Divide</td><td>Aj and Ak Unequal</td><td>More Than 9 Bytes in Buffer</td></tr><tr><td>3</td><td>Aj Source Decimal</td><td>Aj Source Sign Negative</td><td>More Than 9 Bytes in Buffer</td></tr></table>	Bit 1	Bit 2	Bit 3	0	Logical Zero	BDP Busy	Input Complete	1	Aj Source Decimal	Ak Source Decimal	BDP Busy	2	Abort Divide	Aj and Ak Unequal	More Than 9 Bytes in Buffer	3	Aj Source Decimal	Aj Source Sign Negative	More Than 9 Bytes in Buffer
		Decode	Description																						
			<table><tr><th>Bit 1</th><th>Bit 2</th><th>Bit 3</th></tr><tr><td>0</td><td>Logical Zero</td><td>BDP Busy</td><td>Input Complete</td></tr><tr><td>1</td><td>Aj Source Decimal</td><td>Ak Source Decimal</td><td>BDP Busy</td></tr><tr><td>2</td><td>Abort Divide</td><td>Aj and Ak Unequal</td><td>More Than 9 Bytes in Buffer</td></tr><tr><td>3</td><td>Aj Source Decimal</td><td>Aj Source Sign Negative</td><td>More Than 9 Bytes in Buffer</td></tr></table>	Bit 1	Bit 2	Bit 3	0	Logical Zero	BDP Busy	Input Complete	1	Aj Source Decimal	Ak Source Decimal	BDP Busy	2	Abort Divide	Aj and Ak Unequal	More Than 9 Bytes in Buffer	3	Aj Source Decimal	Aj Source Sign Negative	More Than 9 Bytes in Buffer			
		Bit 1	Bit 2	Bit 3																					
		0	Logical Zero	BDP Busy	Input Complete																				
1	Aj Source Decimal	Ak Source Decimal	BDP Busy																						
2	Abort Divide	Aj and Ak Unequal	More Than 9 Bytes in Buffer																						
3	Aj Source Decimal	Aj Source Sign Negative	More Than 9 Bytes in Buffer																						
22	Miscompare Terminate	Terminates input when respective bytes of Aj and Ak source fields miscompare (BDP 3.18, 32).																							
23	No Operation if Ak Length Equals 0	Specifies function being attempted results in no operation if Ak descriptor length equals zero (BDP 3.32).																							
24	Disable PDM	Disables sending BDP PDM Before and After PONR to ICC (BDP 3.33). Also clears Aj Length, Type Register (BDP 3.2).																							
25		Not used.																							
26, 27	Input Select	Indicates which input streams are to be used and synchronized in this operation (BDP 3.11, 32).																							
28	Output Required	<table><tr><th>Decode</th><th>Description</th></tr><tr><td>0</td><td>None.</td></tr><tr><td>1</td><td>Aj source field (A stream).</td></tr><tr><td>2</td><td>Ak source field (B stream).</td></tr><tr><td>3</td><td>Aj and Ak source fields (A and B streams).</td></tr></table>	Decode	Description	0	None.	1	Aj source field (A stream).	2	Ak source field (B stream).	3	Aj and Ak source fields (A and B streams).													
		Decode	Description																						
		0	None.																						
		1	Aj source field (A stream).																						
		2	Ak source field (B stream).																						
3	Aj and Ak source fields (A and B streams).																								
Specifies output is required after Buffer RAM input is complete (BDP 3.14, 18, 19).																									

Table 3-4. BDP Micrand Bits and Descriptions (Sheet 4 of 8)

Bit	Name	Description								
29	Upper Digit Subtract	Produces Complement Upper signal (BDP 3.21) which determines whether Decimal/Binary ALU (BDP 3.12) operates in subtract (set) or add (clear) mode.								
30	Left to Right	Specifies whether byte processing occurs from left to right (set) or right to left (clear) through CM. Determines where sign and slack bytes are located in a string of bytes (BDP 3.5, 7, 8, 10, 19).								
31	Immediate	Specifies that input to A stream is immediate byte from instruction (set) or a byte from CM (clear). Causes continuous A Stream Stage 1 Active (BDP 3.2, 5, 32).								
32	Immediate First Byte	Specifies that input to A stream is for immediate byte followed by fill bytes (set) or for immediate byte repeated continuously (clear). Causes continuous A Stream Stage 1 Last Byte (BDP 3.2, 5).								
33	Clear Signs	Specifies that signs recorded during previous micrand are to be cleared (set) or retained (clear). Causes Negative Sign FF to reset (BDP 3.6, 9).								
34	Enable Arithmetic Overflow	Enables sending BDP Arithmetic Overflow to ICC (BDP 3.33).								
35	Enables Arithmetic Loss of Significance	Enables sending BDP Arithmetic Loss of Significance to ICC (BDP 3.33).								
36	Enables BDP Divide Fault	Enables sending BDP Divide Fault to ICC (BDP 3.33).								
37	Propagate 4-Bit Carry	Enables propagation of digit carry in Decimal/Binary ALU (BDP 3.12, 17).								
38	Propagate 8-Bit Carry	Enables propagation of byte carry in Decimal/Binary ALU (BDP 3.12).								
39	Inhibit Lower-to-Upper Digit Carry	Inhibits carry between lower- and upper-half of 2-digit Decimal/Binary ALU (BDP 3.12).								
40-43	Function A	Expands micrand using Micrand Function A Decoder (BDP 3.1). <table><tr><th>Decode</th><th>Description</th></tr><tr><td>0</td><td>Not used.</td></tr><tr><td>1</td><td>Enable Scale - Enables Shift Left and Shift Right (BDP 3.11).</td></tr><tr><td>2</td><td>Convert - Enables Buffer RAM Address Counter to start decrementing from address of most significant nonzero byte rather than incrementing from address zero. Used for both convert-to-binary and convert-to-decimal operations (BDP 3.14, 19, 22, 24).</td></tr></table>	Decode	Description	0	Not used.	1	Enable Scale - Enables Shift Left and Shift Right (BDP 3.11).	2	Convert - Enables Buffer RAM Address Counter to start decrementing from address of most significant nonzero byte rather than incrementing from address zero. Used for both convert-to-binary and convert-to-decimal operations (BDP 3.14, 19, 22, 24).
Decode	Description									
0	Not used.									
1	Enable Scale - Enables Shift Left and Shift Right (BDP 3.11).									
2	Convert - Enables Buffer RAM Address Counter to start decrementing from address of most significant nonzero byte rather than incrementing from address zero. Used for both convert-to-binary and convert-to-decimal operations (BDP 3.14, 19, 22, 24).									

Table 3-4. BDP Micrand Bits and Descriptions (Sheet 5 of 8)

Bit	Name	Description																												
44-48	Function B	<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>3</td><td>Add - Determines need to complement initial sum in Decimal/Binary ALU (BDP 3.12, 21).</td></tr><tr><td>4</td><td>Subtract - Determines need to complement initial sum in Decimal/Binary ALU (BDP 3.12, 21).</td></tr><tr><td>5</td><td>Collate - Determines compare is made by Register File Compare Control instead of by Decimal/Binary ALU (BDP 3.18).</td></tr><tr><td>6</td><td>Load Table - Specifies that A stream data is to be stored in Register Files A and B (BDP 3.15, 16, 30, 32).</td></tr><tr><td>7</td><td>Not used.</td></tr><tr><td>8</td><td>Numeric Compare - Forces X1 Result Selector status to follow algebraic sign rules (BDP 3.16).</td></tr><tr><td>9</td><td>Not used.</td></tr><tr><td>A</td><td>BDP No Operation - Forces BDP to stop operations immediately (BDP 3.32).</td></tr><tr><td>B</td><td>Unload Decimal - Enables Converted Decimal From Binary/Decimal Converter to destination field through A Stream Stages 1-4, Common Stages 5-7, and C Stream Stages 1-5 (BDP 3.4, 24-26).</td></tr><tr><td>C</td><td>Store Converted Decimal - Produces Complement Upper and Lower signals (BDP 3.21) which determine whether Decimal/Binary ALU (BDP 3.12) operates in subtract (set) or add (clear) mode.</td></tr><tr><td>D</td><td>Not used.</td></tr><tr><td>E</td><td>Not used.</td></tr><tr><td>F</td><td>Not used.</td></tr></table>	<u>Decode</u>	<u>Description</u>	3	Add - Determines need to complement initial sum in Decimal/Binary ALU (BDP 3.12, 21).	4	Subtract - Determines need to complement initial sum in Decimal/Binary ALU (BDP 3.12, 21).	5	Collate - Determines compare is made by Register File Compare Control instead of by Decimal/Binary ALU (BDP 3.18).	6	Load Table - Specifies that A stream data is to be stored in Register Files A and B (BDP 3.15, 16, 30, 32).	7	Not used.	8	Numeric Compare - Forces X1 Result Selector status to follow algebraic sign rules (BDP 3.16).	9	Not used.	A	BDP No Operation - Forces BDP to stop operations immediately (BDP 3.32).	B	Unload Decimal - Enables Converted Decimal From Binary/Decimal Converter to destination field through A Stream Stages 1-4, Common Stages 5-7, and C Stream Stages 1-5 (BDP 3.4, 24-26).	C	Store Converted Decimal - Produces Complement Upper and Lower signals (BDP 3.21) which determine whether Decimal/Binary ALU (BDP 3.12) operates in subtract (set) or add (clear) mode.	D	Not used.	E	Not used.	F	Not used.
		<u>Decode</u>	<u>Description</u>																											
		3	Add - Determines need to complement initial sum in Decimal/Binary ALU (BDP 3.12, 21).																											
		4	Subtract - Determines need to complement initial sum in Decimal/Binary ALU (BDP 3.12, 21).																											
		5	Collate - Determines compare is made by Register File Compare Control instead of by Decimal/Binary ALU (BDP 3.18).																											
		6	Load Table - Specifies that A stream data is to be stored in Register Files A and B (BDP 3.15, 16, 30, 32).																											
		7	Not used.																											
		8	Numeric Compare - Forces X1 Result Selector status to follow algebraic sign rules (BDP 3.16).																											
		9	Not used.																											
		A	BDP No Operation - Forces BDP to stop operations immediately (BDP 3.32).																											
		B	Unload Decimal - Enables Converted Decimal From Binary/Decimal Converter to destination field through A Stream Stages 1-4, Common Stages 5-7, and C Stream Stages 1-5 (BDP 3.4, 24-26).																											
		C	Store Converted Decimal - Produces Complement Upper and Lower signals (BDP 3.21) which determine whether Decimal/Binary ALU (BDP 3.12) operates in subtract (set) or add (clear) mode.																											
		D	Not used.																											
		E	Not used.																											
		F	Not used.																											
		Expands micrand using Micrand Function B Decoder (BDP 3.1).																												
		<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>Not used.</td></tr><tr><td>1</td><td>Round - Forces a carry into the Decimal/Binary ALU if the last byte shifted right end-off of the Aj source Field is greater than four (BDP 3.12).</td></tr><tr><td>2</td><td>Convert to Binary transfer to ALN (BDP 3.0, 24, 32).</td></tr></table>	<u>Decode</u>	<u>Description</u>	0	Not used.	1	Round - Forces a carry into the Decimal/Binary ALU if the last byte shifted right end-off of the Aj source Field is greater than four (BDP 3.12).	2	Convert to Binary transfer to ALN (BDP 3.0, 24, 32).																				
		<u>Decode</u>	<u>Description</u>																											
		0	Not used.																											
		1	Round - Forces a carry into the Decimal/Binary ALU if the last byte shifted right end-off of the Aj source Field is greater than four (BDP 3.12).																											
		2	Convert to Binary transfer to ALN (BDP 3.0, 24, 32).																											

Table 3-4. BDP Micrand Bits and Descriptions (Sheet 6 of 8)

Bit	Name	Description
48, 49	Buffer RAM Input Select	<div> <div>Decode</div> <div>Description</div> </div>
		3 Add or Subtract - Controls decimal complement, recomplement, and sign (BDP 3.12, 21).
		4 Not used.
		5 Two's Complement Compare - Forces constant carry into Decimal/Binary ALU which enables two's complement byte compare (BDP 3.12).
		6 Edit - Enables edit instruction (BDP 3.13, 15, 29-32).
		7 Test Types - Clear BDP Busy after both descriptors are latched during numeric move instruction (BDP 3.11, 32).
		8 Scan - Terminates micrand after Scan Hit during scan instruction (BDP 3.15, 16).
		9 Multiply or Divide - Enables proper destination sign during decimal product or quotient instruction (BDP 3.21).
		A Convert to Decimal - Enables Binary/Decimal Converter for convert-to-decimal operation (BDP 3.19, 24-27).
		B Read Source Data - Enables writing Aj source data into Register Files A and B RAMs during edit instruction (BDP 3.32).
		C Edit Table Load - Enables writing Special Character Table RAM when edit mask is written into Register File A RAM (BDP 3.28, 32).
		D Not used.
		E Not used.
		F Not used.
		Selects input to Common Stage 7 (BDP 3.13).
		<div> <div>Decode</div> <div>Description</div> </div>
		0 Common Stage 6 (normal path).
		1 Register File A RAM (byte translate instruction).
		2 Edit Result Mux (edit instruction).
		3 Not used.

Table 3-4. BDP Micrand Bits and Descriptions (Sheet 7 of 8)

Bit	Name	Description										
50, 51	A Stream Input Select	Selects input to A Stream Stage 1 (BDP 3.5).  <table><tr><th>Decode</th><th>Description</th></tr><tr><td>0</td><td>Not used.</td></tr><tr><td>1</td><td>A Stream Data (normal path).</td></tr><tr><td>2</td><td>Immediate Operand (immediate data instructions).</td></tr><tr><td>3</td><td>Converted Decimal (unload decimal from Binary/Decimal Converter to AC).</td></tr></table>	Decode	Description	0	Not used.	1	A Stream Data (normal path).	2	Immediate Operand (immediate data instructions).	3	Converted Decimal (unload decimal from Binary/Decimal Converter to AC).
Decode	Description											
0	Not used.											
1	A Stream Data (normal path).											
2	Immediate Operand (immediate data instructions).											
3	Converted Decimal (unload decimal from Binary/Decimal Converter to AC).											
52, 53	Table Load Terminate	Controls number of bytes loaded into Register File A/B RAM during load table or read source data operation (BDP 3.15).  <table><tr><th>Decode</th><th>Description</th></tr><tr><td>0</td><td>Load 256 bytes.</td></tr><tr><td>1</td><td>Load 32 bytes (byte scan instruction).</td></tr><tr><td>2</td><td>Load number of bytes specified by value of first byte (edit instruction).</td></tr><tr><td>3</td><td>Not used.</td></tr></table>	Decode	Description	0	Load 256 bytes.	1	Load 32 bytes (byte scan instruction).	2	Load number of bytes specified by value of first byte (edit instruction).	3	Not used.
Decode	Description											
0	Load 256 bytes.											
1	Load 32 bytes (byte scan instruction).											
2	Load number of bytes specified by value of first byte (edit instruction).											
3	Not used.											
54	Wait for Ak Descriptor	Causes A Stream Data to wait until Ak descriptor is latched in BDP before entering A Stream Stage 1 (BDP 3.11, 32).										
55	Calculate Subscript	Enables calculate subscript instruction (BDP 3.4, 23).										
56, 57	Clear Busy	Specifies when BDP Busy to ICP clears (BDP 3.0).  <table><tr><th>Decode</th><th>Description</th></tr><tr><td>0</td><td>Input complete.</td></tr><tr><td>1</td><td>Last byte to AC C stream.</td></tr><tr><td>2</td><td>Last binary byte to ALN.</td></tr><tr><td>3</td><td>Last byte converted to decimal.</td></tr></table>	Decode	Description	0	Input complete.	1	Last byte to AC C stream.	2	Last binary byte to ALN.	3	Last byte converted to decimal.
Decode	Description											
0	Input complete.											
1	Last byte to AC C stream.											
2	Last binary byte to ALN.											
3	Last byte converted to decimal.											
58		Not used.										
59	Point of No Return (PONR)	Indicates that micrand in process has PONR at point where last byte is written into Buffer RAM. Therefore, all processor-detected malfunctions detected in C stream are after PONR (AC 3.6, BDP 3.14, 19, 32, 33).										

Table 3-4. BDP Micrand Bits and Descriptions (Sheet 8 of 8)

Bit	Name	Description
60, 61	Two's Complement Control	Specifies whether two's complement is to be enabled on C stream data from Buffer RAM (BDP 3.21).  <div> <div>Decode</div> <div>Description</div> </div> <div> <div>0</div> <div>No complement.</div> </div> <div> <div>1</div> <div>Complement if Ak output type is 10 or 14 and Aj source is negative.</div> </div> <div> <div>2</div> <div>Complement if Aj input type is 11 or 15 and Aj source is negative.</div> </div> <div> <div>3</div> <div>Complement if Ak output type is 11 or 15 and Aj source is negative.</div> </div>
62	Force Plus One if Aj Negative	Forces plus one on first byte through Decimal/Binary ALU if Aj source data is negative (BDP 3.12).
63	Most Significant Nonzero Byte	Enables loading most significant nonzero byte (clear) or non-FF byte (set) into Buffer RAM Address Counter (BDP 3.14).

ALN Micrand

This 64-bit Functional Unit Micrand provides major cycle control of ALN. A General Micrand FU field of 0 or 1 enables this micrand to execute ALN operations. The ALN micrand is shown in figure 3-17 and described in table 3-5.

Table 3-5. ALN Micrand Bits and Descriptions (Sheet 1 of 9)

Bit	Name	Description
0-6	Constant	Forms literal value used within general network. Field may be used directly as shift count (ALN 3.6), operand for shift count calculations (ALN 3.4), or as adjustment to significance count before exponent arithmetic (ALN 3.7).
7	Response	Causes ALN Go to produce ALN Response unconditionally (ALN 3.24).
8-11	Shift Count Generator Field	Specifies operation to be performed by Shift Count Generator (ALN 3.4). Shift Count Generator Field Decoder (ALN 3.3) provides Shift Counter Generator Control bits to Shift Count Generator.



Table 3-5. ALN Micrand Bits and Descriptions (Sheet 2 of 9)

Bit	Name	Description																																				
		<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>Subtract AC Shift Count Bits 6-11 from Micrand Bits 0-6 using one's complement arithmetic.</td></tr><tr><td>1</td><td>Add AC Shift Count Bits 6-11 to Micrand Bits 0-6.</td></tr><tr><td>2</td><td>Subtract AC Shift Count Bits 6-11 from Micrand Bits 0-6 using one's complement arithmetic. AC Shift Count is sign-extended.</td></tr><tr><td>3</td><td>Add AC Shift Count Bits 6-11 to Micrand Bits 0-6 using ones complement arithmetic. AC Shift Count is sign-extended.</td></tr><tr><td>4</td><td>Subtract AC Shift Count Bits 7-11 from Micrand Bits 0-6, using two's complement arithmetic. AC Shift Count Bit 4 is extended sign.</td></tr><tr><td>5</td><td>Subtract AC Shift Count Bits 6-11 from Micrand Bits 0-6, using two's complement arithmetic. AC Shift Count Bit 4 is extended sign.</td></tr><tr><td>6</td><td>Add zeros to AC Shift Count Bits 0-5.</td></tr><tr><td>7</td><td>Add AC Shift Count Bits 0-5 to AC Shift Count Bits 6-11 using one's complement arithmetic.</td></tr><tr><td>8</td><td>Subtract C170 C Operand Bits 1-15 from C170 B Operand Bits 1-15 using one's complement arithmetic and form absolute value of difference. Shift faults are detected.</td></tr><tr><td>9</td><td>Same as 8 except operands are in C180 format.</td></tr><tr><td>A</td><td>Same as 9 except shift-fault condition is forced for low-order 64 bits of operand to be shifted.</td></tr><tr><td>B</td><td>Subtract C Operand Bits 1-15 from zero-extended Micrand Bits 0-6 using one's complement arithmetic.</td></tr><tr><td>E,F</td><td>Control selection of sign values ALN saves from one major cycle operation to the next.</td></tr></table> <p>Field may be used alternately to control data input to B Register (ALN 3.14) during multiply/divide operations.</p> <table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>Multiply Result.</td></tr><tr><td>8</td><td>B Sign.</td></tr><tr><td>F</td><td>B Operand.</td></tr></table>	<u>Decode</u>	<u>Description</u>	0	Subtract AC Shift Count Bits 6-11 from Micrand Bits 0-6 using one's complement arithmetic.	1	Add AC Shift Count Bits 6-11 to Micrand Bits 0-6.	2	Subtract AC Shift Count Bits 6-11 from Micrand Bits 0-6 using one's complement arithmetic. AC Shift Count is sign-extended.	3	Add AC Shift Count Bits 6-11 to Micrand Bits 0-6 using ones complement arithmetic. AC Shift Count is sign-extended.	4	Subtract AC Shift Count Bits 7-11 from Micrand Bits 0-6, using two's complement arithmetic. AC Shift Count Bit 4 is extended sign.	5	Subtract AC Shift Count Bits 6-11 from Micrand Bits 0-6, using two's complement arithmetic. AC Shift Count Bit 4 is extended sign.	6	Add zeros to AC Shift Count Bits 0-5.	7	Add AC Shift Count Bits 0-5 to AC Shift Count Bits 6-11 using one's complement arithmetic.	8	Subtract C170 C Operand Bits 1-15 from C170 B Operand Bits 1-15 using one's complement arithmetic and form absolute value of difference. Shift faults are detected.	9	Same as 8 except operands are in C180 format.	A	Same as 9 except shift-fault condition is forced for low-order 64 bits of operand to be shifted.	B	Subtract C Operand Bits 1-15 from zero-extended Micrand Bits 0-6 using one's complement arithmetic.	E,F	Control selection of sign values ALN saves from one major cycle operation to the next.	<u>Decode</u>	<u>Description</u>	0	Multiply Result.	8	B Sign.	F	B Operand.
<u>Decode</u>	<u>Description</u>																																					
0	Subtract AC Shift Count Bits 6-11 from Micrand Bits 0-6 using one's complement arithmetic.																																					
1	Add AC Shift Count Bits 6-11 to Micrand Bits 0-6.																																					
2	Subtract AC Shift Count Bits 6-11 from Micrand Bits 0-6 using one's complement arithmetic. AC Shift Count is sign-extended.																																					
3	Add AC Shift Count Bits 6-11 to Micrand Bits 0-6 using ones complement arithmetic. AC Shift Count is sign-extended.																																					
4	Subtract AC Shift Count Bits 7-11 from Micrand Bits 0-6, using two's complement arithmetic. AC Shift Count Bit 4 is extended sign.																																					
5	Subtract AC Shift Count Bits 6-11 from Micrand Bits 0-6, using two's complement arithmetic. AC Shift Count Bit 4 is extended sign.																																					
6	Add zeros to AC Shift Count Bits 0-5.																																					
7	Add AC Shift Count Bits 0-5 to AC Shift Count Bits 6-11 using one's complement arithmetic.																																					
8	Subtract C170 C Operand Bits 1-15 from C170 B Operand Bits 1-15 using one's complement arithmetic and form absolute value of difference. Shift faults are detected.																																					
9	Same as 8 except operands are in C180 format.																																					
A	Same as 9 except shift-fault condition is forced for low-order 64 bits of operand to be shifted.																																					
B	Subtract C Operand Bits 1-15 from zero-extended Micrand Bits 0-6 using one's complement arithmetic.																																					
E,F	Control selection of sign values ALN saves from one major cycle operation to the next.																																					
<u>Decode</u>	<u>Description</u>																																					
0	Multiply Result.																																					
8	B Sign.																																					
F	B Operand.																																					

Table 3-5. ALN Micrand Bits and Descriptions (Sheet 3 of 9)

Bit	Name	Description																								
12-15	Mux Control Field	Specifies operands to be selected by Adder Mux (ALN 3.9) and Shift Network Mux (ALN 3.5). Mux Control Field Decoder (ALN 3.3) provides Mux Control bits to two muxes. Refer to ALN 3.5 and ALN 3.9 diagrams for details of operand selection.																								
16-18	Complement Field	<p>Specifies complement operations to be performed in Adder Mux and Shift Network Mux. Complement Field Decoder (ALN 3.3) provides Complement Bits 7-9 and Complement Control Bits 0-12 to Adder Mux and Shift Network Mux, respectively.</p> <table><tr><th>Decode</th><th>Description</th></tr><tr><td>0</td><td>No complement.</td></tr><tr><td>1</td><td>Complement C Operand Bits 40-63, B Operand Bits 40-63 in Shift Network Mux.</td></tr><tr><td>2</td><td>Complement C Operand, B Operand in Shift Network Mux.</td></tr><tr><td>3</td><td>Complement B Operand in Shift Network Mux and C Operand in Adder Mux.</td></tr><tr><td>4</td><td>Complement negative B Operand and C Operand in Adder Mux. Complement positive B Operand and C Operand in Shift Network Mux.</td></tr><tr><td>5</td><td>Complement negative B Operand and positive C Operand in Adder Mux. Complement positive B Operand and negative C Operand in Shift Network Mux.</td></tr><tr><td>6</td><td>Complement positive B Operand and C Operand in Shift Network Mux.</td></tr><tr><td>7</td><td>Complement B Operand and C Operand in Adder Mux.</td></tr></table> <p>Field may be used alternately to control data input to C Register (ALN 3.13) during multiply/divide operations.</p> <table><tr><th>Decode</th><th>Description</th></tr><tr><td>0</td><td>Quotient or Multiply Result.</td></tr><tr><td>7</td><td>C Operand.</td></tr></table>	Decode	Description	0	No complement.	1	Complement C Operand Bits 40-63, B Operand Bits 40-63 in Shift Network Mux.	2	Complement C Operand, B Operand in Shift Network Mux.	3	Complement B Operand in Shift Network Mux and C Operand in Adder Mux.	4	Complement negative B Operand and C Operand in Adder Mux. Complement positive B Operand and C Operand in Shift Network Mux.	5	Complement negative B Operand and positive C Operand in Adder Mux. Complement positive B Operand and negative C Operand in Shift Network Mux.	6	Complement positive B Operand and C Operand in Shift Network Mux.	7	Complement B Operand and C Operand in Adder Mux.	Decode	Description	0	Quotient or Multiply Result.	7	C Operand.
Decode	Description																									
0	No complement.																									
1	Complement C Operand Bits 40-63, B Operand Bits 40-63 in Shift Network Mux.																									
2	Complement C Operand, B Operand in Shift Network Mux.																									
3	Complement B Operand in Shift Network Mux and C Operand in Adder Mux.																									
4	Complement negative B Operand and C Operand in Adder Mux. Complement positive B Operand and C Operand in Shift Network Mux.																									
5	Complement negative B Operand and positive C Operand in Adder Mux. Complement positive B Operand and negative C Operand in Shift Network Mux.																									
6	Complement positive B Operand and C Operand in Shift Network Mux.																									
7	Complement B Operand and C Operand in Adder Mux.																									
Decode	Description																									
0	Quotient or Multiply Result.																									
7	C Operand.																									
19-22	Exponent Arithmetic Field	Specifies operation to be performed by Exponent Arithmetic Network (ALN 3.8). Exponent Arithmetic Field Decoder (ALN 3.3) sends Exponent Network Control bits to Exponent Arithmetic Network.																								

Table 3-5. ALN Micrand Bits and Descriptions (Sheet 4 of 9)

Bit	Name	Description
		<p><u>Decode</u></p> <p>0 Select B Operand or C Operand, whichever is larger, and remove C170 bias.</p> <p>1 Add Normalize Count and C Operand and insert C170 bias, using one's complement arithmetic.</p> <p>2 Select B Operand or C Operand, whichever is larger.</p> <p>3 Add Normalize Count and C Operand, using two's complement arithmetic.</p> <p>4 Select and complement Normalize Count. Also, specify AND of input operands (bits 0-31) in Large Adder (ALN 3.10).</p> <p>5 Select AC Shift Count Bits 4-15 and insert C170 bias.</p> <p>6 Select C Operand and remove C170 bias.</p> <p>7 Add B Operand and C Operand after removing C170 biases. Use one's complement arithmetic.</p> <p>8 Subtract C Operand from B Operand after removing C170 biases. Use one's complement arithmetic.</p> <p>9 Add B Operand and C Operand after removing C180 bias from C Operand. Use two's complement arithmetic.</p> <p>A Subtract C Operand from B Operand after removing C180 bias from C Operand. Use two's complement arithmetic.</p> <p>B Select Normalize Count and insert C180 bias.</p> <p>C Subtract C Operand from Normalize Count after removing C170 bias from C Operand. Use one's complement arithmetic. Insert C170 bias in result.</p> <p>F Enable parity check of AC Shift Count (ALN 3.24).</p>
23-25	Normalize Encode Control	Governs operations of Normalize Encoder (ALN 3.7). If ALN Functional Unit Micrand bits 54 and 61 are inactive, bits 24 and 25 enable internally generated sign values to be used in subsequent operations.

Table 3-5. ALN Micrand Bits and Descriptions (Sheet 5 of 9)

Bit	Name	Description																		
26-29	Large Adder Control	Normalize Encode Control values specify following operations:																		
		<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>Encode number leading zeros in C Operand Bits 0-63.</td></tr><tr><td>1</td><td>Encode Large Adder (ALN 3.10) overflow.</td></tr><tr><td>2</td><td>Encode Large Adder overflow or number of leading zeros in C Operand Bits 16-63.</td></tr><tr><td>3</td><td>Encode General Network Right Shift 1.</td></tr><tr><td>4</td><td>Encode number of leading zeros in C Operand Bits 16-63.</td></tr><tr><td>5</td><td>Not defined.</td></tr><tr><td>6</td><td>Not defined.</td></tr><tr><td>7</td><td>Not defined.</td></tr></table>	<u>Decode</u>	<u>Description</u>	0	Encode number leading zeros in C Operand Bits 0-63.	1	Encode Large Adder (ALN 3.10) overflow.	2	Encode Large Adder overflow or number of leading zeros in C Operand Bits 16-63.	3	Encode General Network Right Shift 1.	4	Encode number of leading zeros in C Operand Bits 16-63.	5	Not defined.	6	Not defined.	7	Not defined.
		<u>Decode</u>	<u>Description</u>																	
		0	Encode number leading zeros in C Operand Bits 0-63.																	
		1	Encode Large Adder (ALN 3.10) overflow.																	
		2	Encode Large Adder overflow or number of leading zeros in C Operand Bits 16-63.																	
		3	Encode General Network Right Shift 1.																	
		4	Encode number of leading zeros in C Operand Bits 16-63.																	
		5	Not defined.																	
		6	Not defined.																	
7	Not defined.																			
Consists of two two-bit pairs (26, 27 and 28, 29) which calculate Large Adder Result Bits 64-95 and 32-63, respectively (ALN 3.10). Arithmetic operations specified by codes include:																				
<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>Logical OR.</td></tr><tr><td>1</td><td>Addition.</td></tr><tr><td>2</td><td>Logical AND.</td></tr><tr><td>3</td><td>Logical Exclusive OR.</td></tr></table>	<u>Decode</u>	<u>Description</u>	0	Logical OR.	1	Addition.	2	Logical AND.	3	Logical Exclusive OR.										
<u>Decode</u>	<u>Description</u>																			
0	Logical OR.																			
1	Addition.																			
2	Logical AND.																			
3	Logical Exclusive OR.																			
30-32	Large Adder Carry Control	When bits 26-29 specify Large Adder addition, bits 30-32 define whether one's or two's complement arithmetic is to be performed. The field also determines the number bits to be added.																		
		<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>Two's complement 18-bit add.</td></tr><tr><td>1</td><td>One's complement 18-bit add.</td></tr><tr><td>2</td><td>One's complement 60-bit add.</td></tr><tr><td>3</td><td>One's complement 97-bit add.</td></tr><tr><td>4</td><td>Two's complement 64-bit subtract.</td></tr></table>	<u>Decode</u>	<u>Description</u>	0	Two's complement 18-bit add.	1	One's complement 18-bit add.	2	One's complement 60-bit add.	3	One's complement 97-bit add.	4	Two's complement 64-bit subtract.						
		<u>Decode</u>	<u>Description</u>																	
		0	Two's complement 18-bit add.																	
		1	One's complement 18-bit add.																	
		2	One's complement 60-bit add.																	
3	One's complement 97-bit add.																			
4	Two's complement 64-bit subtract.																			

Table 3-5. ALN Micrand Bits and Descriptions (Sheet 6 of 9)

Bit	Name	Description										
		<table><tr><th>Decode</th><th>Description</th></tr><tr><td>5</td><td>Two's complement 32-bit subtract.</td></tr><tr><td>6</td><td>Two's complement 64-bit add.</td></tr><tr><td>7</td><td>Two's complement 32-bit add.</td></tr></table>	Decode	Description	5	Two's complement 32-bit subtract.	6	Two's complement 64-bit add.	7	Two's complement 32-bit add.		
Decode	Description											
5	Two's complement 32-bit subtract.											
6	Two's complement 64-bit add.											
7	Two's complement 32-bit add.											
33	Plus 1	Used conditionally by Large Adder during two's complement 32- or 64-bit addition. Plus 1 is added to adder inputs if C Operand Sign is negative.										
34-37	ALN Output Mux Control	Select ALN Result in ALN Output Mux (ALN 3.10, 3.12). Bits 34, 35 select result bits 0-15; bits 36, 37 select bits 16-63.										
		<table><tr><th>Decode</th><th>Description</th></tr><tr><td>0</td><td>Shift Network Output.</td></tr><tr><td>1</td><td>Large Adder Result Bits 32-95.</td></tr><tr><td>2</td><td>Large Adder Result Bits 0-47, Normal Exponent Bits 0-7, Exponent Bits 8-15.</td></tr><tr><td>3</td><td>Compare Code 0, 1; Encase Exponent Bits 0-7.</td></tr></table>	Decode	Description	0	Shift Network Output.	1	Large Adder Result Bits 32-95.	2	Large Adder Result Bits 0-47, Normal Exponent Bits 0-7, Exponent Bits 8-15.	3	Compare Code 0, 1; Encase Exponent Bits 0-7.
Decode	Description											
0	Shift Network Output.											
1	Large Adder Result Bits 32-95.											
2	Large Adder Result Bits 0-47, Normal Exponent Bits 0-7, Exponent Bits 8-15.											
3	Compare Code 0, 1; Encase Exponent Bits 0-7.											
38	Double Precision	Used as general purpose bit to perform any of following operations: <ul style="list-style-type: none"><li>• Copy branch condition met (inactive ALN Unbranch) from previous micrand to Compare Code (ALN 3.20).</li><li>• Enable C180 double-precision floating-point loss-of-significance detection (ALN 3.21).</li><li>• Enable detection of C170 integer multiplication (ALN 3.22).</li><li>• Force floating-point operand selection if zero exponent (nonstandard) is detected (ALN 3.23).</li><li>• Enable clearing of Micrand Register if C170 Indefinite or Infinite is detected (ALN 3.24).</li></ul>										
39	Enable Exponent Underflow, Overflow	Permits appropriate endcase result to be generated if exponent underflow or overflow is detected by Exponent Arithmetic Network. Endcase exponent is selected in ALN Output Mux, Bits 0-15 (ALN 3.12).										
40-42	Exception Selection Field	Decode enables detection of selected exception (ALN 3.33).										

Table 3-5. ALN Micrand Bits and Descriptions (Sheet 7 of 9)

Bit	Name	Description																		
		<table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>No exception enabled.</td></tr><tr><td>1</td><td>Enable Large Adder overflow.</td></tr><tr><td>2</td><td>Enable multiply overflow.</td></tr><tr><td>3</td><td>Enable divide overflow.</td></tr><tr><td>4</td><td>Enable floating-point loss of significance.</td></tr><tr><td>5</td><td>Enable arithmetic loss of significance.</td></tr><tr><td>6</td><td>Enable divide fault.</td></tr><tr><td>7</td><td>Enable unbranch.</td></tr></table>	<u>Decode</u>	<u>Description</u>	0	No exception enabled.	1	Enable Large Adder overflow.	2	Enable multiply overflow.	3	Enable divide overflow.	4	Enable floating-point loss of significance.	5	Enable arithmetic loss of significance.	6	Enable divide fault.	7	Enable unbranch.
<u>Decode</u>	<u>Description</u>																			
0	No exception enabled.																			
1	Enable Large Adder overflow.																			
2	Enable multiply overflow.																			
3	Enable divide overflow.																			
4	Enable floating-point loss of significance.																			
5	Enable arithmetic loss of significance.																			
6	Enable divide fault.																			
7	Enable unbranch.																			
43	C170 Mode	Indicates C170 format in General and Multiply/Divide networks.																		
44	Enable Endcase Condition	Enables Endcase Select Decoder (ALN 3.19) and shift fault detection (ALN 3.4,23).																		
45-47	Endcase/Branch Condition Select Field	If Micrand Bit 44 is active, decode of field is interpreted as follows (ALN 3.19):  <table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>Force endcase condition if endcase is detected during multiplication.</td></tr><tr><td>1</td><td>Force endcase condition if endcase is detected during division.</td></tr><tr><td>2</td><td>Force endcase condition if endcase detected during addition.</td></tr><tr><td>3</td><td>Force endcase condition if endcase is detected during subtraction.</td></tr><tr><td>4</td><td>Retain endcase detected by previous micrand.</td></tr><tr><td>5</td><td>Interpret B Operand Bits 62, 63 as least significant bits of j and C Operand Bit 32 as <math>X_k</math> Register Bit 32. Perform enter signs per j function.</td></tr><tr><td>6</td><td>Perform mark to boolean function.</td></tr><tr><td>7</td><td>Perform last cycle of endcase operation.</td></tr></table>	<u>Decode</u>	<u>Description</u>	0	Force endcase condition if endcase is detected during multiplication.	1	Force endcase condition if endcase is detected during division.	2	Force endcase condition if endcase detected during addition.	3	Force endcase condition if endcase is detected during subtraction.	4	Retain endcase detected by previous micrand.	5	Interpret B Operand Bits 62, 63 as least significant bits of j and C Operand Bit 32 as $X_k$ Register Bit 32. Perform enter signs per j function.	6	Perform mark to boolean function.	7	Perform last cycle of endcase operation.
<u>Decode</u>	<u>Description</u>																			
0	Force endcase condition if endcase is detected during multiplication.																			
1	Force endcase condition if endcase is detected during division.																			
2	Force endcase condition if endcase detected during addition.																			
3	Force endcase condition if endcase is detected during subtraction.																			
4	Retain endcase detected by previous micrand.																			
5	Interpret B Operand Bits 62, 63 as least significant bits of j and C Operand Bit 32 as $X_k$ Register Bit 32. Perform enter signs per j function.																			
6	Perform mark to boolean function.																			
7	Perform last cycle of endcase operation.																			

Table 3-5. ALN Micrand Bits and Descriptions (Sheet 8 of 9)

Bit	Name	Description																		
		<p>If Micrand Bits 40-42=7, decode of Micrand Bits 45-47 has following meaning (ALN 3.20):</p> <table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>Branch if B Operand equals C Operand.</td></tr><tr><td>1</td><td>Branch if B Operand is not equal to C Operand.</td></tr><tr><td>2</td><td>Branch if B Operand is greater than C Operand in C180 mode or if B Operand is less than C Operand in C170 mode.</td></tr><tr><td>3</td><td>Branch if B Operand is greater than or equal to C Operand.</td></tr><tr><td>4</td><td>Branch if C Operand exponent is in range in C170 mode or if condition specified by j is met in C180 mode.</td></tr><tr><td>5</td><td>Branch if C Operand exponent is out of range in C170 mode or if B Operand Bits 20-31 do not equal C Operand Bits 20-31 in C180 mode.</td></tr><tr><td>6</td><td>Branch if C Operand exponent is definite in C170 mode.</td></tr><tr><td>7</td><td>Branch if C Operand exponent is indefinite in C170 mode.</td></tr></table>	<u>Decode</u>	<u>Description</u>	0	Branch if B Operand equals C Operand.	1	Branch if B Operand is not equal to C Operand.	2	Branch if B Operand is greater than C Operand in C180 mode or if B Operand is less than C Operand in C170 mode.	3	Branch if B Operand is greater than or equal to C Operand.	4	Branch if C Operand exponent is in range in C170 mode or if condition specified by j is met in C180 mode.	5	Branch if C Operand exponent is out of range in C170 mode or if B Operand Bits 20-31 do not equal C Operand Bits 20-31 in C180 mode.	6	Branch if C Operand exponent is definite in C170 mode.	7	Branch if C Operand exponent is indefinite in C170 mode.
<u>Decode</u>	<u>Description</u>																			
0	Branch if B Operand equals C Operand.																			
1	Branch if B Operand is not equal to C Operand.																			
2	Branch if B Operand is greater than C Operand in C180 mode or if B Operand is less than C Operand in C170 mode.																			
3	Branch if B Operand is greater than or equal to C Operand.																			
4	Branch if C Operand exponent is in range in C170 mode or if condition specified by j is met in C180 mode.																			
5	Branch if C Operand exponent is out of range in C170 mode or if B Operand Bits 20-31 do not equal C Operand Bits 20-31 in C180 mode.																			
6	Branch if C Operand exponent is definite in C170 mode.																			
7	Branch if C Operand exponent is indefinite in C170 mode.																			
48	Enable Floating-Point Compare	Indicates floating-point comparison is being performed. (ALN 3.20).																		
49, 50	Shift Count Select	<p>Select shift count value to enter Shift Network (ALN 3.6).</p> <table><tr><th><u>Decode</u></th><th><u>Description</u></th></tr><tr><td>0</td><td>Micrand Bits 1-6.</td></tr><tr><td>1</td><td>Significance Count.</td></tr><tr><td>2</td><td>Shift Count.</td></tr><tr><td>3</td><td>Shift Count.</td></tr></table> <p>Micrand Bit 49 also denotes unsigned multiplication in Multiply/Divide Network (ALN 3.13, 3.14).</p>	<u>Decode</u>	<u>Description</u>	0	Micrand Bits 1-6.	1	Significance Count.	2	Shift Count.	3	Shift Count.								
<u>Decode</u>	<u>Description</u>																			
0	Micrand Bits 1-6.																			
1	Significance Count.																			
2	Shift Count.																			
3	Shift Count.																			

Table 3-5. ALN Micrand Bits and Descriptions (Sheet 9 of 9)

Bit	Name	Description
51-53	Round Control	Inserts round bit in bit position to right of least significant bit of operand. Micrand Bit 51 forces Shift Network Mux Bit 64 to a one (ALN 3.5). Micrand Bit 52 activates same bit if B Operand Bit 0 is not equal to B Operand Bit 16 and C Operand Bit 0 is not equal to C Operand Bit 16, or if operand signs are not equal. Micrand Bit 53 causes round bit to be inserted at Adder Mux Bit 48 (ALN 3.9). Bit 53 also enters Multiply/Divide Network to enable BDP convert-to-binary operations (ALN 3.15) or to enable insertion of round bits into Divide Network (ALN 3.17).
54	Multiply/Divide Start	Specifies first micrand of multiply or divide operation, and permits current micrand to be used as Major Cycle Control (ALN 3.2).
55-59	Soft Control Entry Address	Determines entry point in Soft Control Memory RAM (ALN 3.1) for four-word soft control sequence during multiply or divide operation.
60	Shift Network Zero Extend	Determines whether zeros or sign is extended in high-order bit positions of operand shifted by Shift Network (ALN 3.6). In Multiply/Divide Network, bit indicates integer divide (ALN 3.17) or acts as next byte request during BDP convert-to-binary operations (ALN 3.2).
61	Multiply/Divide Run	Specifies second or subsequent micrand of multiply/divide operation and permits previous micrand to be used as Major Cycle Control (ALN 3.2).
62	Enable 18-Bit Compare	Used as a general purpose bit to perform one of the following operations: <ul style="list-style-type: none"> <li>• Enables 18-bit comparisons (ALN 3.20).</li> <li>• Enables floating-point operand selection if zero (nonstandard) exponent is detected (ALN 3.23).</li> <li>• Indicates 64-bit integer multiplication (ALN 3.18).</li> </ul>
63	Force Zero Exponent	Enables zero exponent during floating-point double-precision calculations when significant data is not present at Normalize Encoder and double-precision loss of significance has not been detected (ALN 3.21).

#### INSTRUCTION CONTROL PIPELINE (ICP 1.0)

ICP contains the facilities to control instruction execution in conjunction with CST, OPI, and a designated executing Functional Unit or units (AC, SM, LM, BDP, or ALN).

The interaction of ICP with the units mentioned above and the significance of Ranks 13, 22, 32, 41 and 50 are described in section 1 and elsewhere in part 2 of this section. What remains for discussion are the specific contents of the pipeline and pipeline control logic.



## P REGISTERS

P Registers exist in all five ranks of ICP. In each rank the P Register contains the byte number portion of the Process Virtual Address (PVA) for the instruction in that rank. The PVA for a given instruction enters ranks 32 and 41 each time a micrand associated with the instruction enters one of those ranks. The PVA is latched in rank 50 at the point of no return (PONR) only if there is no pending unbranch or interrupt.

Rank 22 P enters OPI during operations which require the program address for particular computations. The C180 relative branch (2E) instruction, for example, creates a branch address from P and the contents of an X Register.

Rank 41 P may enter the Rank 50 UTP Register in the event a micrand in Rank 41 has addressed memory and an addressing error or exception has been detected.

IF uses Rank 50 P to compute the address of the instruction following a conditional branch instruction in an RNI sequence when the branching condition has not been met (unbranch).

## IMMEDIATE OPERAND, j AND k REGISTERS

Rank 13, 22, 32, and 41 registers contain immediate operands and operating register designators. Rank 13 j and k designators enter j, k Incrementers to produce j+1 and k+1 for C180 double-precision floating-point operations.

Rank 22 j, k Operand Select bits and Rank 22 Immediate Operand bits supply register designators and immediate operands to OPI. C170 designators include j and k. C180 designators include i, j, and k. K is the C170 immediate operand. C180 immediate operands include jk, Q, and D. Immediate operands and j, k designators are redefined as L (length) and 0 (offset) for BDP descriptors.

When an A Register in the Register File supplies a memory address that fails, Rank 41 j, k Operand Select accompany Recovery Address into the Rank 50 UTP Register to designate which A Register was the source of the error. This information is used to determine the PVA of the associated instruction.

## INSTRUCTION CONTROL SIGNAL REGISTERS

Instruction Control Signal Registers contain control signals from IF and exception and error flags activated in various stages of instruction assembly and execution.

The C170i register designator (alternately the BDP descriptor type field) enters the pipeline at Rank 13. Additional signals from IF entering the Instruction Control Signal Registers include instruction address out of range, address specification error, opcode hit, debug profile, IF-detected malfunction, and page table search without find.

Signals from SM and AC enter the Rank 41 Instruction Control Signal Register to indicate addressing and security violations as well as other conditions.

Accumulated exceptions, errors and control signals proceed from pipeline ranks 41 and 50 to ICC for evaluation. Errors and exceptions may become the basis for initiating interrupt routines in ICC.

### Debug Profile

In C180 mode a hardware feature exists which aids in testing and debugging software. When the system enables this feature, a predetermined set of conditions (Debug Mask) are compared with a given instruction's Debug Profile. If one or more Debug Mask conditions is detected in the instruction's Debug Profile, ICP Potential Debug Hit activates in ICP Debug Hit Control.

ICP Potential Debug Hit enables a debug microcode routine. The routine determines whether the instruction enters predefined memory areas and whether it enters for purposes specified by debug codes from the job's debug list. Typically, the Debug Mask is compiled from all debug codes contained in a debug list.

If a debug hit occurs, a trap interrupt takes place. The specific operations from which debug hits may result include instructions reading or writing in memory, branching or issuing a call to memory, or fetching another instruction.

### Largest Ring

During an operation in which the A Register is to be written, SM sends the largest available ring number to the Rank 41 Instruction Control Signal Register. ICP sends this value to OPI. OPI makes a final determination of the ring value to be placed in the A Register.

### GENERAL MICRAND REGISTERS

After the General Micrand enters the General Micrand Register (CST 1.0), the Functional Unit and General Micrand extension codes (General Micrand bits) proceed directly to the Rank 32 General Micrand Register. The General Micrand Register also sends point of no return and clock condition register bits. The Register File write address for a given micrand enters the Rank 32 General Micrand Register after being selected in OPI.

The Functional Unit and General Micrand extension codes enter Functional Unit Go Control and Response Control, respectively, from Rank 32.

Rank 32 Register Data Select Write Bits 37 through 39 return to OPI to control, among other operations, the number of bytes written into a selected Register File location.

Rank 41 General Micrand Bits 24, 25 control selection of data in the DAI Register (OPI 1.0).

Rank 50 Register Data Select Write bits enter OPI to control data entry into live registers.

Rank 50 General Micrand bits enter ICC to assist in processing any exceptions detected during micrand execution.

### PIPE VALID CONTROL

Pipe Valid Control indicates when a given rank contains valid information. It plays a primary role in controlling movement of micrands and instructions through the pipeline. Rank 13, 22, and 32 Enable, Functional Unit Go (Rank 41), and Rank 50 P Register Enable permit loading of the ranks specified.

An instruction remains in rank 22 during the course of its micrand sequence. Functional unit Go signals control micrand movement from rank 32 to rank 41. They also permit CST Rank 22 Valid to gate a new micrand from rank 22 to rank 32 after the preceding micrand has entered rank 41. Rank 50 P Register Enable, required to load the micrand into Rank 50, activates when Rank 41 is valid and the micrand's result is not being inhibited.

When the last micrand associated with the rank 22 instruction produces an Exit in CST and enters rank 32, a new instruction enters rank 22 from rank 13.

The rank 12 instruction enters rank 13 with the arrival of Rank 12 Valid from IF. At the same time, CST controls the advance of other instructions in the IF buffer ranks.

#### FUNCTIONAL UNIT GO RESPONSE CONTROL

General Micrand bits entering rank 32 of the pipeline generate a functional unit Go signal to indicate the functional unit in which the associated Functional Unit Micrand will execute. Response and busy signals from the functional units enter ICP to advance micrands/ instructions in the pipeline.

Segment Descriptor Fetch Control operations are described in section 5.

#### OPERAND ISSUE (OPI 1.0)

This section of the IU contains the Register File, which consists of 64 64-bit memory locations. The Register File holds data for use by the functional units during Functional Unit Micrand execution. Fifty-two of the 64 memory locations are written during exchange sequences; the rest are written during system initialization.

Under General Micrand control, OPI addresses the Register File and determines read and write data paths. Read and write accesses required 16 ns. Three Register File read operations and one write operation may occur in a given 64-ns major cycle.

In addition, OPI provides the following:

- Live Registers, which contain values needed for program execution or for monitoring CP performance. Refer to section 1 for more information about Live Register characteristics.
- Immediate Operand Select network, which supplies data from an instruction or other source (excluding the Register File) to the executing functional unit.
- Conflict-checking hardware which enables shortstop data paths for enhanced CP performance.
- Conflict-checking hardware that delays micrand movement in the pipeline until operands modified by one micrand become available to a second one.
- Counters and comparators used to track multiple-word memory accesses such as exchange operations, calls, returns, and load/store multiple-word operations.

## GENERAL MICRAND CONTROL

In the online mode of operation, the General Micrand controls reading and writing of operands in OPI. The General Micrand fields entering OPI include the Write Data Select (WDS), Immediate Data Select (IDS), and Register/Data Select (RDS) a, b, and c fields. In addition, RDSd from the AC Functional Unit Micrand may be used to supply a Register File write address. IDS controls data entry into the AC Immediate Data and AC Address Offset muxes. WDS controls selection of data entering the DAI Register. All other major read and write operations in OPI are determined by the RDS fields. Refer to CST description earlier in this section for details of IDS, WDS, and RDS formats and content.

As the supplier of operands to executing functional units, OPI may select up to three values from the Register File or other source. When ALN or BDP is the executing functional unit, RDSb and RDSd may each supply one read operand. When AC is the executing functional unit, OPI uses RDSa, b, and c to supply as many as three read operands.

In addition to reading operands, a General Micrand may perform a single write operation in the Register File or a Live Register. When ALN or BDP is the executing Functional Unit, RDSa designates the Register File or Live Register write address of ALN Result. (The result of an executing BDP Functional Unit Micrand typically comes from ALN.) RDSd normally designates the Register File write address of a word from memory. A Register File write address enters the Minipipe before functional unit micrand execution. RDSa's write address also enters the instruction pipeline to control Live Register writes. The reappearance of the write address from these delay circuits coincides with the result's arrival in OPI.

Figures 3-18 and 3-19 show address and data paths in OPI and the executing functional unit during typical ALN and memory operations. Note the role of the General Micrand fields within OPI.

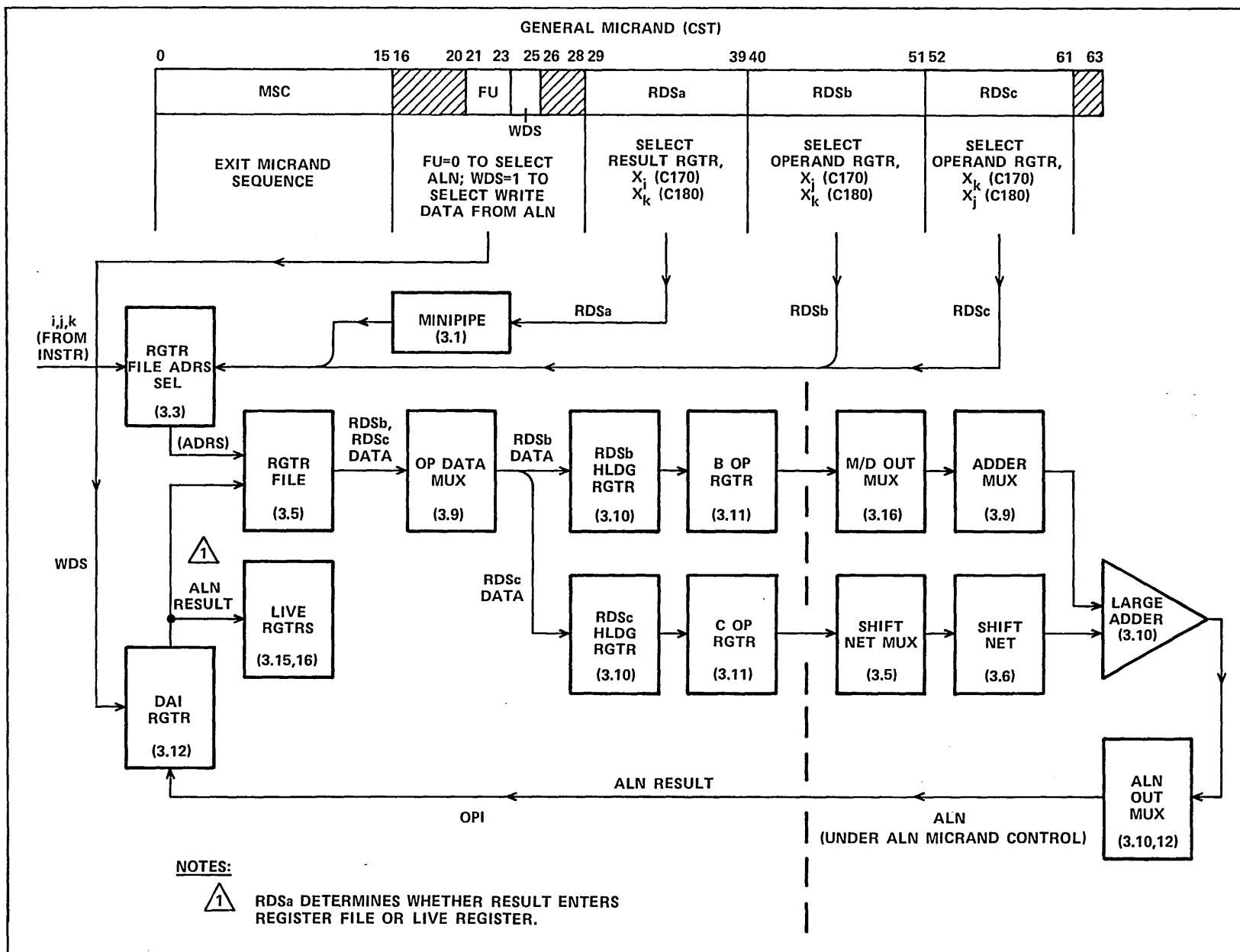
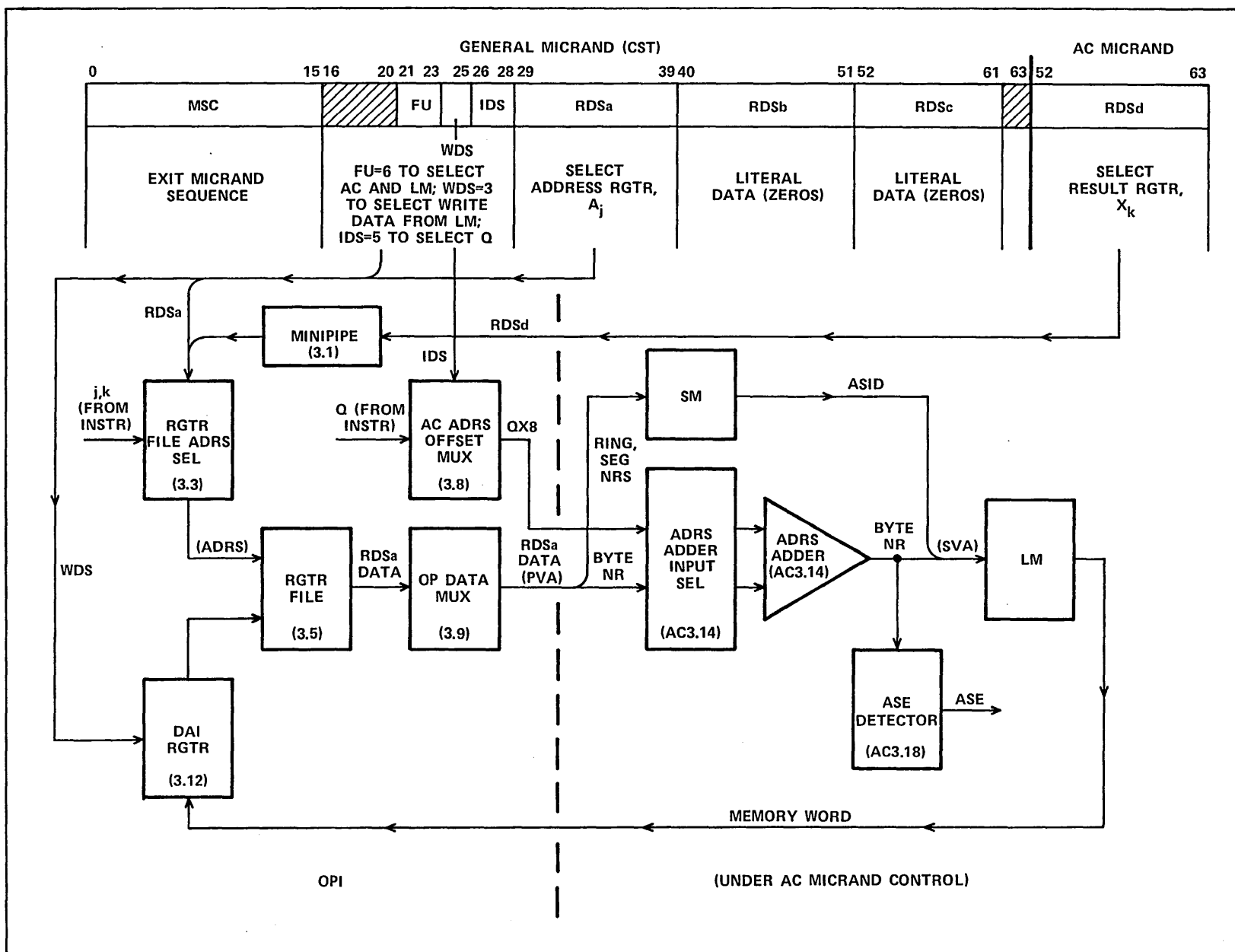


Figure 3-18. Integer Sum/Difference (C170 36, 37, C180 24, 25) Instructions



## OPI READS, WRITES

A General Micrand's RDS fields enter the Register File Address Select Network at micrand execution Time 22. The RDS read fields (a, b, and c) enter the Micrand RDS Field Selector. Write Field RDSa or RDSd (from the Functional Unit Micrand) enters the RDS Write Selector before proceeding to the instruction pipeline as RDS Write Select and to the Micrand RDS Field Selector as RDSw.

The Micrand RDS Field Selector selects the RDS fields consecutively on minor cycle boundaries. It selects RDSa between Time 22 and Time 23, RDSb between Time 23 and Time 30, RDSd between Time 30 and Time 31, and RDSw between Time 31 and Time 32.

## Read Operations

How the RDS read fields control OPI hardware from the time of their selection is based on how the fields are configured. There are five RDS read formats. Any combinations of these formats may be present in a General Micrand's RDSa, b, and c fields.

- Register Addressing Function (RAF) - If the RD and LT bits in the RDS field are zero, RAF selects a register designator as the Register File read address.
- Literal Register Address (LRA) - If the RD bit is a zero and the LT bit a one, LRA and BI subfields become the Register File read address.
- Literal Data (LD) - If the RD bit is a one and LO bit is a zero, LD becomes Immediate Operand Register Bits 57 through 63 in Immediate Operand Muxes.
- Data Subfunction Select (DSS) - If the RD bit is a one, the LO bit a one, and the DL bit a zero, DSS selects a single or multiple operand field from the instruction or N Register value as the Immediate Operand Mux output.
- Live Register Read (LRR) - If the RD bit, LO bit, and DL bit equal 1, LRR selects Live Register Read Data in the Live Register Read Mux.

## Register Addressing Function (RAF)

The LT bit and RAF subfield select the appropriate Register File read address in the Register File Address Selector. Selectable values include Rank 22 Designators from ICP, C180i from ICP, and the contents of the A Start and X Start registers. The Register File Address enters the Register File at Time 23 if selected by RDSa, Time 30 if selected by RDSb, and Time 31 if selected by RDSd. Register File Read Data is available to the Operand Data Mux one minor cycle after it is addressed. RDSd Register File Read Data is active for two minor cycles.

## Literal Register Address (LRA)

The LT bit selects the LRA and BI subfields as the literal Register File Address in Register File Address Select. The Register File Address enters the Register File according to the schedule described for RAF read addresses. Register File Read Data is available to the Operand Data Mux one minor cycle after the data is addressed.

### Literal Data (LD)

Inactive LO selects the LD subfield in the Immediate Operand Muxes. The RDSa LD subfield enters the mux at Time 23. RDSb's LD subfield enters at Time 30, and RDSc's enters at Time 31. Each value arrives at the Operand Data Mux one minor cycle after reaching the Immediate Operand Muxes. These times correspond to the arrival of Register File Read Data at the Operand Data Mux.

### Data Subfunction Select (DSS)

LO, inactive DL, and the DSS subfield select the appropriate Rank 22 Immediate Operand or Rank 22 P in the Immediate Operand Muxes. An RDS field's Immediate Operand enters the Operand Data Mux according to the schedule described for Literal Data.

### Live Register Read (LRR)

The LRR subfield (RDS Bits) selects the appropriate output from the Live Register Read Mux. RDSa's Live Register Read Data enters the Immediate Operand Mux at Time 23, RDSb's enters at Time 30, and RDSc's enters at Time 31. Live Register Read Data reaches the Operand Data Mux as Immediate Operand Register bits one minor cycle later (concurrent with the other read operands already described).

### Operand Registers

Read operands selected by RDSa, b, and c pass through the Operand Data Mux during consecutive minor cycle intervals (from Time 30 to 31, Time 31 to 32, Time 32 to 33). If an RDS read field specifies RAF or LRA with zeros in four least significant bit positions and an active UZ bit, the Operand Data Mux substitutes zeros for the value actually read from the Register File.

When Operand Data is a CM address (A Register value), the most significant 32 bits contain the segment number and the ring number being sent to SM for evaluation. The ring number first passes through the Largest Ring Selector, a delay path. The byte number portion of the same address enters the AC Address Formation Network as A/B Operand after it passes through the Operand Left Shift Network.

Thirty-two bit operands and byte addresses selected by RDSa or RDSb to form a memory address in AC share the A/B Operand path. The RDSa data is present in AC from Time 30 to 31, the RDSb data, from Time 31 to 32.

The Operand Left Shift Network may shift the 32 least significant Operand Data bits selected by RDSa or RDSb an amount determined by the DS subfield.

If a hold condition occurs because the micrand in ICP rank 32 is unable to enter rank 41 to execute, address formation data recirculates through the Operand Data Mux from the RDSa and RDSb Holding Registers until it can be processed in AC.

When three values are used by the AC Address Formation Network in C170 mode, the least significant 21 bits of the RDSb read operand enter the B Operand Save Register at Time 32. B Operand Save Register bits become AC Address Offset for AC Address calculations.



When Operand Data is a 64-bit data word (instead of address argument) selected by RDSb or RDSc, the least significant 32 bits pass through the Operand Left Shift Network. The 32 most significant bits bypass the shifter, with four bits following a default path through the Largest Ring Selector.

RDSb read data enters the RDSb Holding Register at Time 32. RDSc read data enters the RDSc Holding Register at Time 33.

At Time 40 RDSb and RDSc read operands enter the B Operand Register and the C Operand Register, respectively. The B Operand enters ALN for arithmetic operations. The C Operand is either an ALN input operand, data word to be stored in memory by way of AC, or A or B stream length designator for BDP operations.

### Write Operations

An RDS write field (RDSa or RDSd) has one of three formats and designates whether the Register File or a Live Register is to be entered.

- Register Addressing Function (RAF) - If the RD and LT bits in the RDS field are zero, RAF selects an instruction register designator to be the Register File write address. WW subfield determines which byte pairs are to be written.
- Literal Register Address (LRA) - If the RD bit is a zero and the LT bit a one, LRA and BI subfields become the Register File write address. WW subfield determines which byte pairs are to be written.
- Live Register Write/Complete Cycle Control (LRW/CSC) - If the RD bit is a one, LRW/CSC field determines which Live Register is written or which micrand completion control activity takes place.

### Register Addressing Function (RAF)

The RDS Write Selector selects RDSa from the General Micrand or RDSd from the Function Unit Micrand to become RDSw. The WW subfield proceeds to ICP from the RDS Write Selector as RDS Write Select. From Time 31 to Time 32 the Micrand RDS Field Selector selects RDSw to enter the Register File Address Selector.

The RDSw LT bit and RAF subfield select the appropriate Register File write address in the Register File Address Selector. Selectable values include Rank 22 Designators from ICP, Cl80i from ICP, and the contents of the A Start and X Start registers.

Register File Address Index 0 or 1 enters the Minipipe Input at Time 32 and Minipipe Rank 33 at Time 33. The Register File address proceeds to Minipipe Rank 40 and 50 before reentering the Register File Address Selector between Time 41 and Time 52 as Rank 50 Write Address. The Register File Address addresses the Register File at Time 52.

Register File Write Data enters the byte positions designated by the WW subfield, which is delayed in ICP. Register File Write Data reaches the Register File by way of the DAI Register (which it enters at Time 50). The General Micrand's WDS Field, also delayed in ICP, controls write data selection at the input to the DAI Register. DAI Register Bits 16 through 19 pass through the Largest Ring Selector before entering the Register File. When the WW subfield equals 1 or 3, OPI selects the largest ring assignable to an address being stored in the Register File.

#### Literal Register Address (LRA)

The RDS Write Selector selects RDSw as described in the RAF discussion. The WW subfield proceeds to ICP as RDS Write Select. From Time 31 to Time 32 the Micrand RDS Field Selector selects the BI and LRA subfields to enter the Minipipe Input as RDS Bits at Time 32. The Register File write address arrives in Minipipe Rank 33 at Time 33 and proceeds to Minipipe Ranks 40 and 50. Rank 50 Write Address enters the Register File Address Selector between Time 51 and Time 52. The write address addresses the Register File at Time 52.

The entry of write data into the Register File byte positions designated by the WW subfield is the same as for RAF write operations.

#### Live Register Write/Complete Cycle Control (LRW/CSC)

The RDS Write Selector selects RDSa from the General Micrand or RDSd from the Functional Unit Micrand to become RDS Write Select. RDS Write Select enters ICP before returning to the OPI Live Registers as Rank 50 RDS Write at Time 50. The LRW/CSC code contained in Rank 50 RDS Write selects the Live Register to be loaded with write data from the DAI Register. The General Micrand's WDS field controls write data selection at the input to the DAI Register. The register selected or the end of cycle action undertaken is indicated in the Live Register Write Decoder (OPI 3.15). SIT and PIT are among the devices that can be written under Live Register write control. The DAI Register path through the Stream Mode Mux supplies write data to the various mask and condition registers in ICC, which likewise are subject to Live Register write control.

Figure 3-20 shows the timing for reading operands and storing results when an ALN operation immediately precedes a memory access in a micrand sequence.

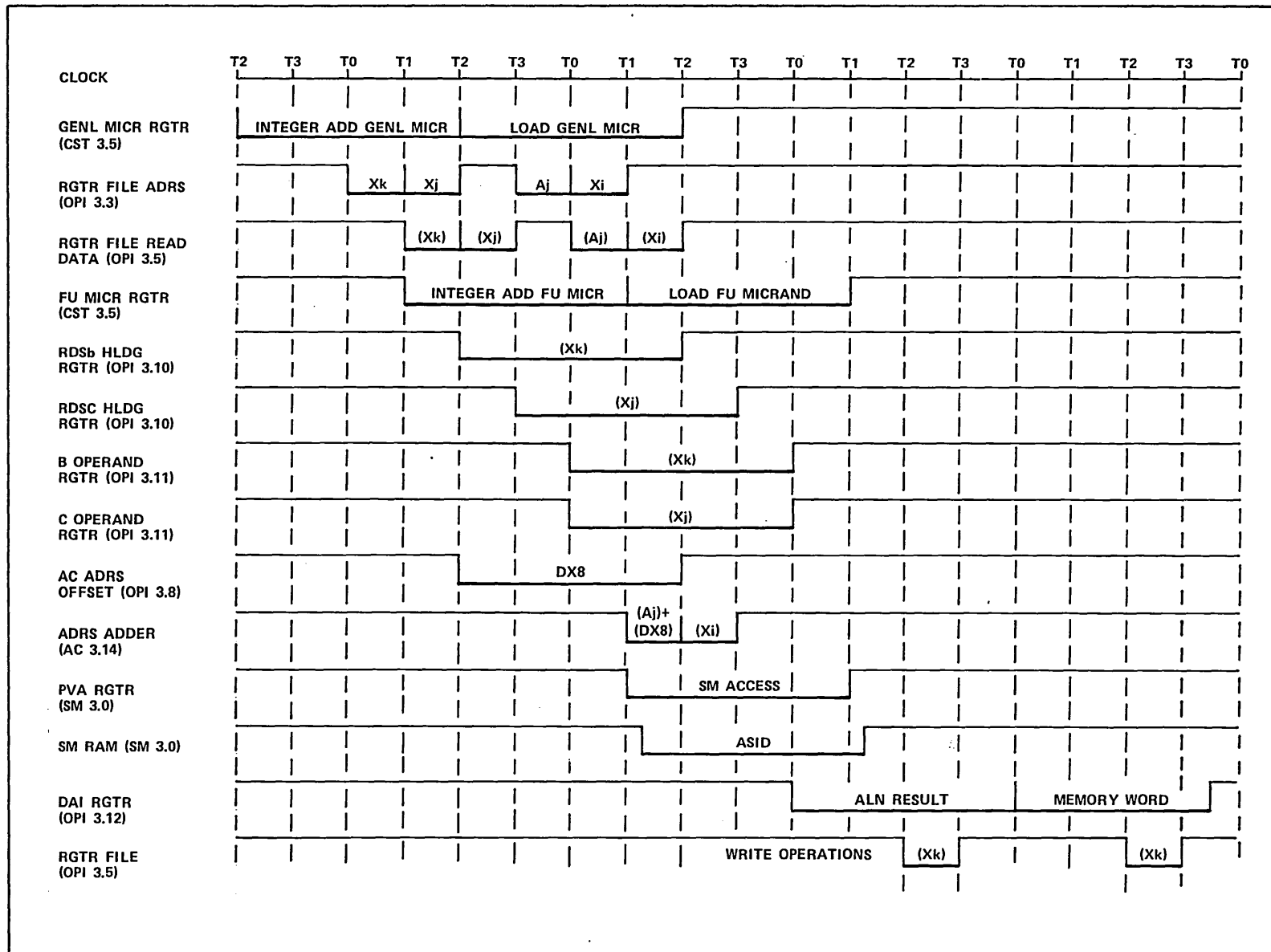


Figure 3-20. Operand Access by C180 Integer Sum (24) and C180 Load Word (A2) Instructions

## OPERAND CONFLICTS, SHORTSTOPPING

Because of the delay between reading of operands and storing results in the Register File, operand conflicts between micrands may occur. When one micrand requires a read operand from the Register File that another micrand is in the process of modifying, the operand addressed in the Register File is not valid. Such conflicts may arise between two micrands that are adjacent in the pipeline or between micrands that are separated by a single micrand.

These conflicts may be resolved in one of two ways. The first micrand's result may be shortstopped to join other read operands selected by the micrand which caused the conflict. If the conflict cannot be resolved directly by shortstopping, the micrand requiring the result stalls in ICP Rank 22 until it is possible to obtain the read operand via the shortstop path. Two shortstop paths are available. One path is to the inputs of the B Operand Register and C Operand Register. The other path is from the DAI Register to the Operand Data Mux.

Conflicts are detected in the Operand Conflict Detectors from Minipipe (rank 23) and Minipipe Rank 33, 40, and 50 bit inputs. Minipipe (rank 23) bits are the Register File read addresses of the micrand in ICP Rank 22. Minipipe Rank 33, 40, and 50 bits are the Register File write addresses for the micrands in ICP ranks 32 through 50. Comparisons between the read addresses and the various write addresses establish whether conflicts exist. The read addresses specified by RDSa, b, and c enter the detectors at Time 23, Time 30, and Time 31. Minipipe Input selects the RDSw Register File address at Time 32. At Time 33 the write address enters Minipipe Rank 33.

The conditions under which shortstop paths resolve conflicts are as follows:

- Condition 1 If the Register File read address entering the Operand Conflict Detectors at Time 31 equals the Minipipe Rank 40 write address, Operand Shortstop enables result data directly to the C Operand Register.
- Condition 2 If the Register File read address entering the Operand Conflict Detectors at Time 30 equals the Minipipe Rank 40 address, and the read data is not going to AC and does not require ring checking or shifting, Operand Shortstop enables incoming result data directly to the B Operand Register.
- Condition 3 If the Register File read address entering the Operand Conflict Detectors at Time 23 equals the Minipipe Rank 40 write address and ICP Rank 50 is about to become valid (response is about to advance the rank 41 micrand), incoming result data enters the Operand Data Mux via the DAI Register. Here it becomes the RDSa read operand. This circuit is the Register File Shortstop path.
- Condition 4 If the Register File read address entering the Operand Conflict Detectors at Time 30 equals the Minipipe Rank 50 write address, incoming result data takes the Register File Shortstop path to become the RDSb read operand.
- Condition 5 If the Register File read address entering the Operand Conflict Detectors at Time 31 equals the Minipipe Rank 50 write address, incoming result data takes the Register File Shortstop path to become the RDSc read operand.

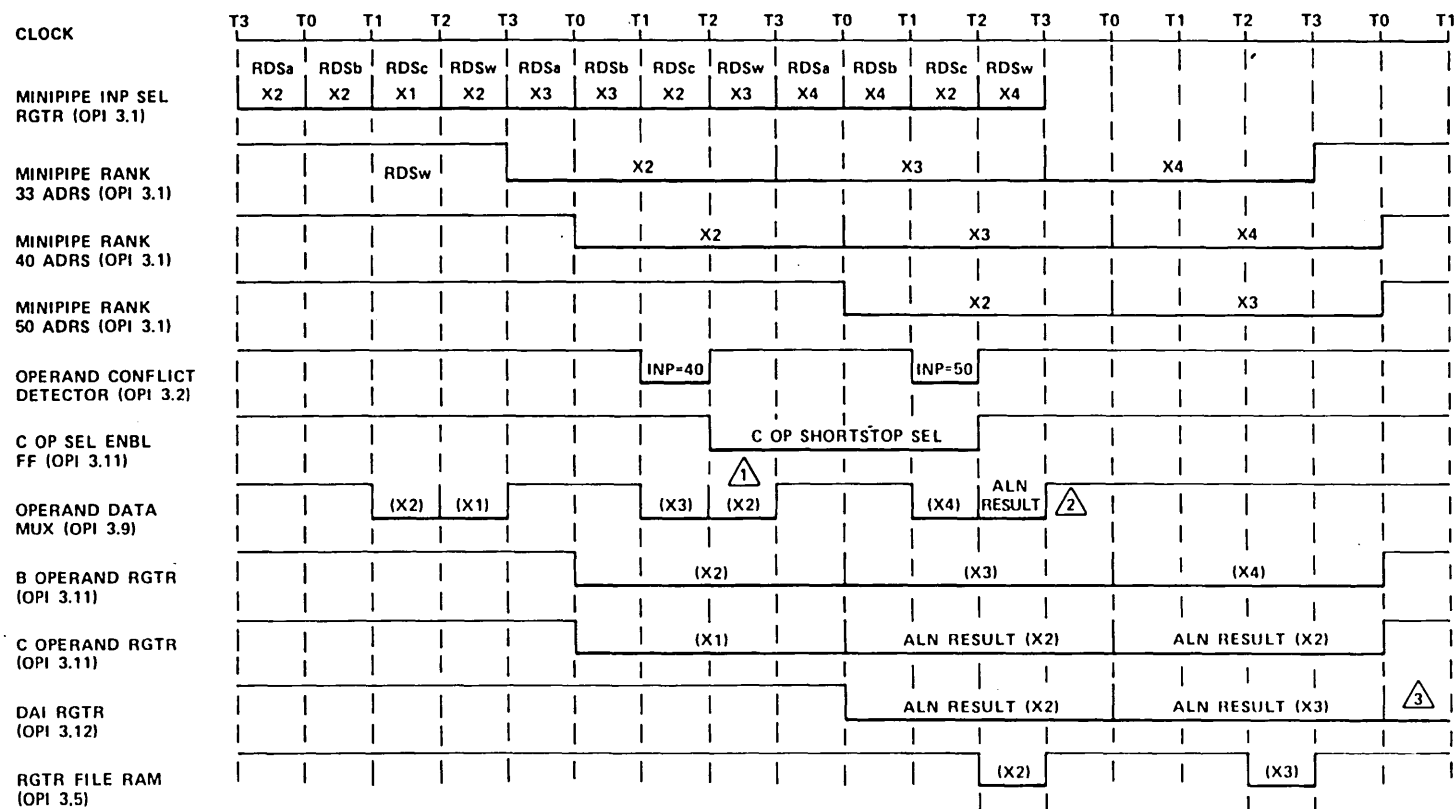
The conditions under which pipeline delays and shortstop paths together resolve conflicts are as follows:

- Condition 6 If the Register File read address entering the Operand Conflict Detectors at Time 23 equals the Minipipe Rank 33 write address and the read data is going to AC, A Operand Conflict stalls the micrand in ICP Rank 22 for one major cycle. The Condition 3 Register File Shortstop operation then takes place.

- Condition 7 If the Register File read address entering the Operand Conflict Detectors at Time 23 equals the Minipipe Rank 40 write address but the Rank 50 valid condition is inhibited, A Operand conflict stalls the micrand in ICP Rank 22 until a response gates the micrand out of ICP Rank 41. The Condition 3 Register File Shortstop operation then takes place.
- Condition 8 If the Register File read address entering the Operand Conflict Detectors at Time 30 equals the Minipipe Rank 40 write address, and the read operand is going to AC, needs shifting, or is to be ring-checked in OPI, B Operand Conflict stalls the micrand in ICP Rank 22 for one major cycle. The Condition 4 Register File Shortstop operation then takes place.

Read operands specified by RDSc to go to AC are not needed in AC until after Time 40. Pipeline delays are not required to resolve conflicts involving these operands.

Figure 3-21 demonstrates the timing of Operand Shortstop and Register File Shortstop operations for consecutive micrands executing in ALN.



## NOTES:

- 1 INVALID DATA. VALID (X2) ENTERS C OPERAND RGTR DIRECTLY FROM ALN.
- 2 DAI REGISTER SUPPLIES ALN RESULT.
- 3 ALN RESULT (X4).

Figure 3-21. Operand Conflict Timing for Three Consecutive Integer Sum Instructions ( $X1+X2 \rightarrow X2$ ,  $X2+X3 \rightarrow X3$ ,  $X2+X4 \rightarrow X4$ )

## EXCHANGE OPERATIONS

There are two principal categories of CP operations. One is designated C180 job (or user) mode and the other, C180 monitor (or system) mode. An exchange typically switches the CP between one mode and the other by substituting a new CP instruction sequence. To change instruction sequences requires switching their respective exchange packages in the Register File.

The C180 operating system regards all C170 job or monitor mode operations as C180 job mode routines. In C180 job mode, the CP executes C180 and C170 exchange instructions and enables exchange interrupts. Exchange interrupts occur when exception conditions are detected which require handling by a monitor routine.

In C180 monitor mode, the CP executes C180 exchange instructions, but does not enable exchange interrupts.

The C180 Exchange (02) instruction switches the instruction execution sequence from C180 job to C180 monitor mode, or from C180 monitor to C180 job or C170 job/monitor mode. Unlike C170 exchanges (which use a single system-designated memory space), C180 exchange operations use two memory spaces, one for each exchange package. The Register File contains the starting addresses for both memory spaces.

C170 exchange instructions include the Central Exchange Jump to (Bj) + K (013jk), Central Exchange Jump to MA (013XX), and the IOU Exchange Jump (0026) instructions. The C170 monitor mode 013jk instruction exchanges to a C170 job mode instruction sequence, which exchanges back to the monitor mode at its conclusion. The 013XX C170 job mode instruction exchanges to a C170 monitor mode sequence, which exchanges back to the job mode when it ends. Either of these instructions takes precedence over an IOU Exchange Jump request if they occur simultaneously.

The IOU Exchange Jump instruction permits the IOU to initiate exchange operations in the CP. Two versions of the instruction involve exchanges from C170 job to C170 monitor mode. The third does not involve a change of mode.

The Exchange Code indicates the type of C170 IOU Exchange Jump being requested. Depending on the version of the instruction, the memory address of an exchange package may be MA (contained in the current exchange package) or a value the IOU calculates itself (Exchange Address). When Exchange Address is calculated, it enters the AC Address Formation Network (AC 1.0) as a Live Register read operand.

Exchange interrupt sequences switch the CP operation from C180 job or C170 job/monitor mode to C180 monitor mode. Setting a condition bit in ICC typically initiates an exchange interrupt, provided the corresponding mask bit is set. The hardware disables subsequent exchange interrupts while the system remains in C180 monitor mode.

C180 call and return instructions involve partial exchanges between the Register File and Stack Frame Save Areas in memory. (Unlike an exchange package memory location, a Stack Frame Save Area is part of a user's reserved memory space.) Calls and returns do not change the C180 job or monitor classification of the CP operation. Calls and returns may occur within C180 job or monitor mode or may proceed between C180 job and C170 job/monitor when used in conjunction with a C170 Trap 180 (017) instruction. A trap interrupt is a specialized call operation caused by detection of an exception condition which invokes a trap handler routine.

During C180 02 instructions or exchange interrupts, the exchange operation first stores the exchange package contained in the Register File and Live Registers in memory. Then it writes a new exchange package into the Register File and Live Registers. During C170 exchange instructions, the operation takes place in reverse order, with scratch registers in the Register File used as interim storage areas. The use of a single storage area in memory requires that information be transferred to OPI before the Register File's exchange package enters the reserved memory space.

The C180 exchange package consists of 52 64-bit words, containing the information needed by the CP to execute the instruction sequence associated with it. Figure 3-22 shows the C170 and C180 exchange packages used during C180 02 instructions or exchange interrupts. The C170 exchange package used during 13 and 26 instructions contains 16 64-bit words.



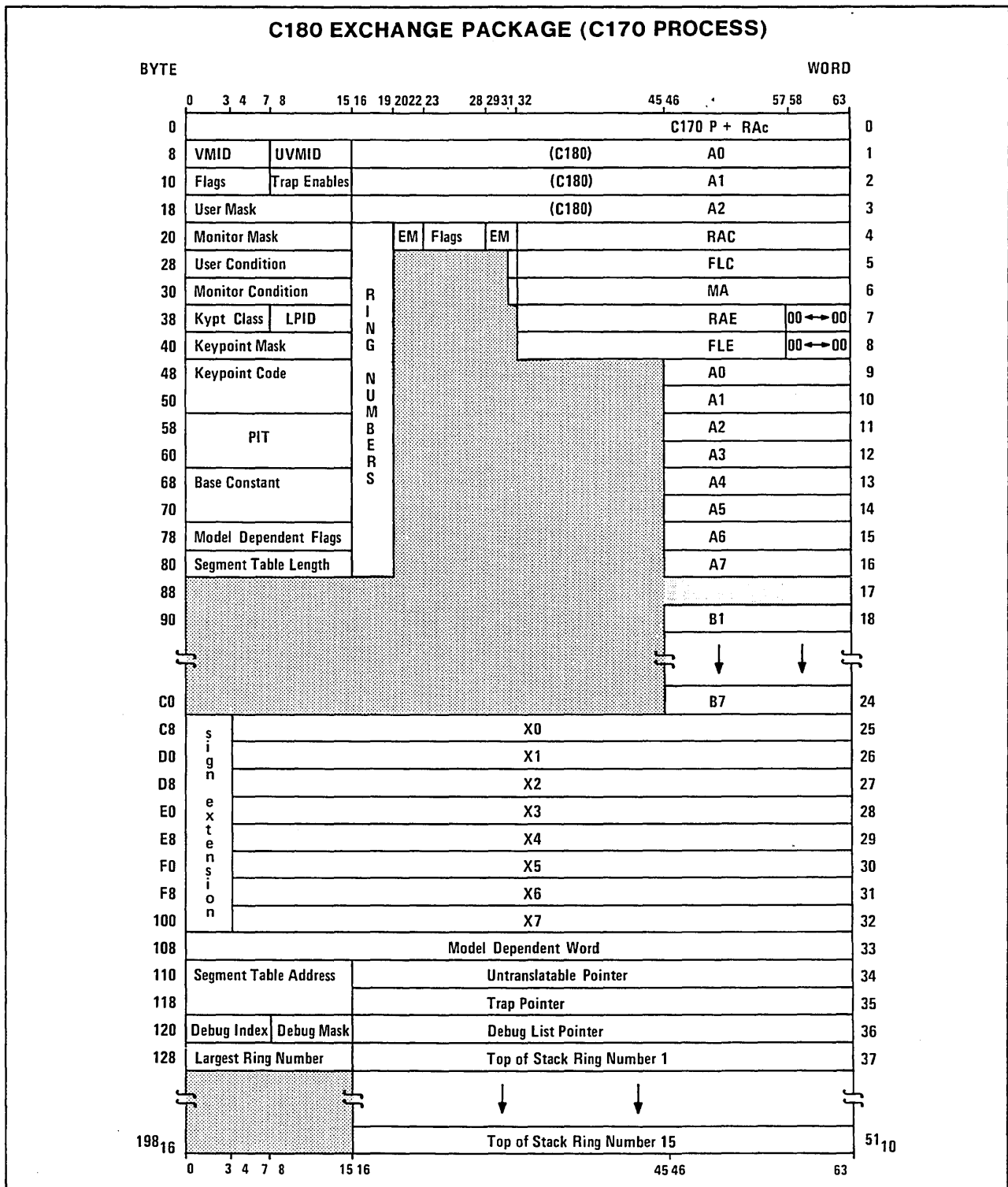


Figure 3-22. C170, C180 Exchange Packages (Sheet 1 of 2)

# C180 EXCHANGE PACKAGE (C180 PROCESS)

BYTE					WORD	
	0	7	8	15	16	63
0	P					0
8	VMID	UVMID		A0		1
10	Flags	Trap Enables		A1		2
18	User Mask		A2			3
20	Monitor Mask		A3			4
28	User Condition		A4			5
30	Monitor Condition		A5			6
38	Kypt Class	LPID		A6		7
40	Keypoint Mask		A7			8
48	Keypoint Code		A8			9
50			A9			10
58	PIT		AA			11
60			AB			12
68	Base Constant		AC			13
70			AD			14
78	Model Dependent Flags		AE			15
80	Segment Table Length		AF			16
88	X0					17
90	X1					18
≈						≈
C0						24
C8	X8					25
D0	X9					26
D8	XA					27
E0	XB					28
E8	XC					29
F0	XD					30
F8	XE					31
100	XF					32
108	Model Dependent Word					33
110	Segment Table Address		Untranslatable Pointer			34
118			Trap Pointer			35
120	Debug Index	Debug Mask		Debug List Pointer		36
128	Largest Ring Number		Top of Stack Ring Number 1			37
≈						≈
						↓
						↓
198			Top of Stack Ring Number 15			51
16						10
0	0	7	8	15	16	63

Figure 3-22. C170, C180 Exchange Packages (Sheet 2 of 2)

When the exchange package is transferred from OPI to memory, individual data words proceed to AC as C Operands. The Register File Address through most of the sequence comes from the A Start Counter or the X Start Counter. The counters initially are loaded from the C Operand Register and increment each major cycle to read consecutive data words from the Register File. The micrand controlling a specific phase of the transfer exits when the contents of a counter equals the contents of the corresponding terminate register.

When the exchange package is transferred to OPI from memory, the contents of the X Start Counter indicates the Register File write address. The counter increments each major cycle, sending a new address to the Register File Address Selector. Each address goes from the Register File Address Selector to the Minipipe Input as Register File Address Index 1. The address becomes Stream Mode Tag Bits, which proceed from Minipipe Rank 33 to LM and CMC. CMC retains the tag bits to accompany LM Read Data to OPI as the write address.

After LM Read Data arrives at the DAI Register, it enters the Register File and associated Live Registers. The write address, CMC Tag, enters the Register File Address Selector and Live Register load control to indicate the storage location.

Table 3-6 shows the contents of OPI counters and registers while a C180 exchange Register File load operation is taking place. At any point during the sequence, N indicates how many write addresses have been formed by the X Start Counter. X Start is compared to X Terminate to determine the end of a particular micrand loop.

Table 3-6. C180 Exchange Counter Operations

PRIOR TO MICRAND EXECUTION			AFTER MICRAND EXECUTION			
MICRAND (CST 3.5)	X TERMN (OPI 3.14)	X START (OPI 3.14)	X START (OPI 3.14)	N (OPI 3.14)	<sup>1</sup> RGTR FILE ADDRESS	RGTR CONTENTS
1 ↓ 1 EXIT	F <sub>16</sub> ↓ F X START = X TERMN	0 1 2 3 4 5 6 7 8 9 A B C D E F	1 2 3 4 5 6 7 8 9 A B C D E F 0	1 2 3 4 5 6 7 8 9 A B C D E F 10	10 11 12 13 14 15 16 17 18 19 1A 1B 1C 1D 1E 1F	A0, VMID, UVMID A1, FLAGS, TRAP ENBLS A2, UMR A3, MMR A4, UCR A5, MCR A6, KEYPOINT CLASS A7, KEYPOINT MASK A8, KEYPOINT CODE A9, KEYPOINT CODE AA, PIT AB, PIT AC, CONSTANT AD, CONSTANT AE, FLAGS AF, STL
2 ↓ 2 EXIT	F ↓ F X START = X TERMN	0 ↓ 1-E ↓ F	1 ↓ 2-F ↓ 0	11 ↓ 12-1F ↓ 20	20 ↓ 21-2E ↓ 2F	X0 ↓ X1-XE ↓ XF
3 4 5 6	F ↓ F	0 1 2 3	1 2 3 4	21 22 23 24	0B 0C 0D 0E	MODEL DEPENDENT WORD STA, UTP STA, TRAP POINTER DEBUG LIST POINTER

NOTES:



MEMORY TAG ADDRESSES REGISTER FILE. TAG IS PRODUCED IN OPI FROM X START AND CONSTANT, SENT TO CMC, AND RETURNED TO OPI WITH DATA. MICRANDS 3-6 DO NOT USE X START DATA.

## Register File Maps

Figure 3-23 shows the contents of Register File locations 10-2F<sub>16</sub> during C170 mode operations. Figure 3-24 shows the contents of the Register File when configured for C180 mode operations. Locations 0-F<sub>16</sub> are the the same for C170 and C180 mode operations. Locations 30-3F<sub>16</sub> are scratch registers reserved for C170 exchange operations.

RGTR	0	78	15 16 19 20	31 32	45 46	57 58	63
10	VMID	UVMID	C180 A0				
11	FLAGS	TRAP ENBLS	C180 A1				
12	USER MASK		C180 A2				
13	MON MASK		RING NUM BERS	△1	RAC		
14	USER COND			△2	FLC		
15	MON COND			△3	MA		
16	KYPT CLASS	LPIC		RAE			000000
17	KYPT MASK			FLE			000000
18	KYPT CODE (UPPER)						C170 A0
19	KYPT CODE (LOWER)						C170 A1
1A	PIT (UPPER)						C170 A2
1B	PIT (LOWER)						C170 A3
1C	BASE CONSTANT (UPPER)						C170 A4
1D	BASE CONSTANT (LOWER)					C170 A5	
1E	MODEL DEPENDENT FLAGS					C170 A6	
1F	SEG TABLE LENGTH					C170 A7	

NOTES:

△1

NOT USED  
UEM ENABLE FLAG  
ESM MODE FLAG  
BLOCK COPY FLAG  
SOFTWARE FLAG  
INSTR STACK PURGE FLAG  
SOFTWARE FLAG  
EXIT MODE-INDEFINITE  
-INFINITE  
-AOR

△2

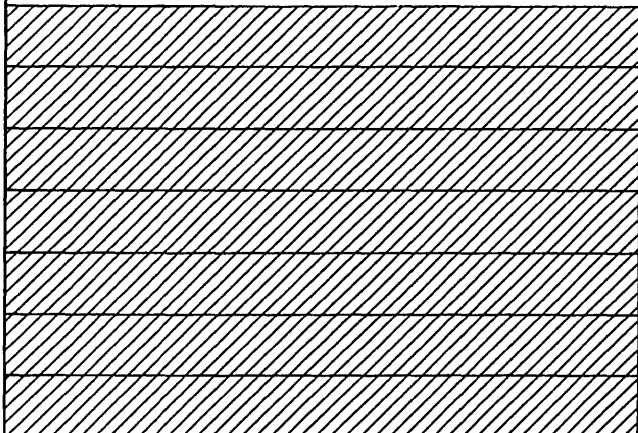








C170 MONITOR FLAG

△3

EXIT MODE HALT

BITS 20-22  
BIT 23  
BIT 24  
BIT 25  
BIT 26  
BIT 27  
BIT 28  
BIT 29  
BIT 30  
BIT 31  
BIT 31  
BIT 31

Figure 3-23. C170 Register File Map (Sheet 1 of 2)

RGTR	0	3	4	45	46	63
20	C170 SCRATCH REGISTER					
21						C170 B1
22						C170 B2
23						C170 B3
24						C170 B4
25						C170 B5
26						C170 B6
27						C170 B7
28		C170 X0				
29		C170 X1				
2A		C170 X2				
2B		C170 X3				
2C		C170 X4				
2D		C170 X5				
2E		C170 X6				
2F		C170 X7				

NOTE:

 SIGN EXTENSION

Figure 3-23. C170 Register File Map (Sheet 2 of 2)



RGTR	0	15 16	31 32	63
00	C180 SCRATCH REGISTER			
01	MAC DATA ASSEMBLY REGISTER (USED BY ENTRY 32 <sub>16</sub> )			
02	PAGE SIZE MASK	A <sub>j</sub> HOLDING REGISTER		
03	C 0 0 0 <sub>16</sub>	A <sub>k</sub> HOLDING REGISTER		
04	0 0 0 0	PTL	PAGE TABLE ADDRESS	
05	P I D			MON PROCESS STATE POINTER
06	006F <sub>16</sub>	0000	JOB PROCESS STATE POINTER	
07	TOP OF STACK INDEX			CURRENT PROCESS POINTER
08	C180 SCRATCH REGISTER 1			
09	C180 SCRATCH REGISTER 2			
0A	F F F F F F F F F F F F F F F F <sub>16</sub>			
0B	MODEL DEPENDENT WORD			
0C	S T A (UPPER)	UNTRANSLATABLE POINTER		
0D	S T A (LOWER)	TRAP POINTER		
0E	DEBUB IDX, DEBUG MASK	DEBUG LIST POINTER		
0F <sub>16</sub>	C180 SCRATCH REGISTER			

Figure 3-24. C180 Register File Map (Sheet 1 of 4)

RGTR	0	7 8	15 16	63
10	VMID	UVMID	A0 (DYNAMIC SPACE POINTER)	
11	FLAGS	TRAP ENABLES	A1 (CURRENT STACK FRAME POINTER)	
12	USER MASK		A2 (PREVIOUS SAVE AREA POINTER)	
13	MON MASK		A3 (BINDING SECTION POINTER)	
14	USER COND		A4 (ARGUMENT POINTER)	
15	MON COND		A5	
16	KYPT CLASS	LPID	A6	
17	KYPT MASK		A7	
18	KYPT CODE (UPPER)		A8	
19	KYPT CODE (LOWER)		A9	
1A	PIT (UPPER)		AA	
1B	PIT (LOWER)		AB	
1C	BASE CONSTANT (UPPER)		AC	
1D	BASE CONSTANT (LOWER)		AD	
1E	MODEL DEPENDENT FLAGS		AE	
1F <sub>16</sub>	SEG TABLE LENGTH		AF	

Figure 3-24. C180 Register File Map (Sheet 2 of 4)



RGTR	0	63
20		X0
21		X1
22		X2
23		X3
24		X4
25		X5
26		X6
27		X7
28		X8
29		X9
2A		XA
2B		XB
2C		XC
2D		XD
2E		XE
2F	16	XF

Figure 3-24. C180 Register File Map (Sheet 3 of 4)

RGTR	0	63
30	NATIVE C170 EXCHANGE WORKING REGISTER	
31	NATIVE C170 EXCHANGE WORKING REGISTER	
32	NATIVE C170 EXCHANGE WORKING REGISTER	
33	NATIVE C170 EXCHANGE WORKING REGISTER	
34	NATIVE C170 EXCHANGE WORKING REGISTER	
35	NATIVE C170 EXCHANGE WORKING REGISTER	
36	NATIVE C170 EXCHANGE WORKING REGISTER	
37	NATIVE C170 EXCHANGE WORKING REGISTER	
38	NATIVE C170 EXCHANGE WORKING REGISTER	
39	NATIVE C170 EXCHANGE WORKING REGISTER	
3A	NATIVE C170 EXCHANGE WORKING REGISTER	
3B	NATIVE C170 EXCHANGE WORKING REGISTER	
3C	NATIVE C170 EXCHANGE WORKING REGISTER	
3D	NATIVE C170 EXCHANGE WORKING REGISTER	
3E	NATIVE C170 EXCHANGE WORKING REGISTER	
3F <sub>16</sub>	NATIVE C170 EXCHANGE WORKING REGISTER	

Figure 3-24. C180 Register File Map (Sheet 4 of 4)

#### ADDRESS OFFSET

The IDS field from the Rank 22 General Micrand controls data selection in the AC Immediate Data Mux and the AC Address Offset Mux. AC Address Offset, typically an immediate operand or a B Operand, is used in memory address computations by the AC Address Control Network.

## TIMERS

### Process Interval Timer (PIT)

PIT is a 32-bit counter that decrements every microsecond. It measures time intervals within an executing program and sets a bit in the UCR (ICC 1.0) when the count reaches zero.

### System Interval Timer (SIT)

SIT is a 32-bit counter that decrements every microsecond. SIT measures time intervals within a monitor mode program. When the count reaches zero, a bit sets in the MCR (ICC 1.0).

## OFFLINE MAC OPERATIONS

MAC may read or write the Register File when the CP is offline. Refer to section 2 for details of offline Register File read and write operations.



#### SECTION 4

#### EXECUTING FUNCTIONAL UNITS



PART 1

ARITHMETIC AND LOGICAL NETWORK (ALN)





The following text is to be read in conjunction with ALN Level 1 and 3 diagrams for an explanation of ALN operations.

The Arithmetic and Logical Network (ALN) contains hardware required to execute all floating-point and integer add, subtract, multiply, and divide instructions found in the C170 and C180 instruction sets. ALN also processes C170 and C180 shift and boolean instructions, as well as C170 pack/unpack, normalize, and population count instructions. It plays a role in branch instructions, evaluating branch conditions. ALN performs operations during most other C170 and C180 instructions, passing operands between Register File locations, comparing operands, or completing other required arithmetic steps.

ALN performs operations on values supplied by OPI or BDP. All results return to OPI except for exception conditions which are sent to ICP. ALN consists of a Multiply/Divide Network and a general network. Negative C170 operands typically are represented in one's complement form, negative C180 operands in two's complement form. Operands may be 18 or 60 bits for C170 operations, or 32 or 64 bits for C180 operations.

#### ALN CONTROL

The ALN Functional Unit Micrand and Soft Control Data Output control ALN operations when ALN Go and ALN Request are present.

ALN Go generates ALN Response to indicate the current ALN operation is being completed. ALN usually performs unconditionally, because it has few significant interdependencies with other functional units. ALN Response follows ALN Go's arrival in ALN, provided ALN Functional Unit Micrand Bit 7 is set. Bit 7 is always set except when BDP data is expected. Convert Response indicates the end of a BDP convert decimal-to-binary operation. Because ALN performs a portion of the conversion with a multiply algorithm, ALN delays ALN Response until Convert Response arrives from BDP.

All general network operations occur on 64-ns boundaries with General Network Control determining the nature of the operation. Fields from the ALN Functional Unit Micrand and control bits from the Micrand Decoders enter the General Network Control/Micrand Register. The decoders expand ALN Functional Unit Micrand control beyond what 64 incoming bits could accomplish if they passed directly into the general network.

Multiply/Divide Network operations are controlled by 20 Functional Unit Micrand bits and Soft Control Data Output. Four soft control words are issued at a minor cycle rate for each arriving Functional Unit Micrand during multiplication or division. The Functional Unit Micrand supplies the starting address in Soft Control Memory.

Like ALN Go, ALN Request results from a translation in ICP of the General Micrand's functional unit field. ALN Request gates Soft Control Data Output and various Functional Unit Micrand bits into Multiply/Divide Control Register. From there Multiply/Divide Control governs multiply and divide operations.

MAC loads Soft Control Memory with data from the MCU during system initialization. Refer to Soft Control Memory Write in section 2 for details.

Various detected ALN Errors enter the PDM Tester to activate ALN PDM. ICC senses ALN PDM in its Monitor Control Register (MCR).

## ALN CONTROL FIELD DESCRIPTIONS

ALN Functional Unit Micrand bit and field descriptions are shown in section 3, figure 3-17.

Refer to the Maintenance Aids section of the Maintenance and Parts Data manual for a map of ALN Soft Control Memory bits described here.

<u>Bit</u>	<u>Definition</u>
0	Enable iteration result and shifted multiplier path into C Register (3.13).
1	Load C Register (3.13).
2	Multiply round bit (3.15). Enable dividend into Dividend Register (3.17).
3	Load Partial Summing Network Sum, Carry registers (3.15). Enables B, C operand sign to Divide Network (3.13,14). Enable B operand to Quotient Register (3.17).
4	Enable Partial Summing Network feedback path (3.15). Enable divide iteration results into Dividend Register (3.17).
5	Enable Pop Counter increment (3.16). Disable forced select of subtracter result (3.17).
6	Enable C Register parity check (3.13). Latch multiply/divide result sign (3.14).
7	Load new soft control word (3.2).
8	Select Remainder Register Least Significant Bit input (3.17).
9	Divide round bit (3.17).
10	Enable quotient bit select (3.17). Enable Pop counter Load (3.16). Enable divide Quotient Register (3.17).
11	Enable Multiply Final Adder carry errors out (3.16).
11-13	Multiply/divide output mux control code: (3.16).  0 - Select B operand, 2 - Select C Register. 3 - Select pop count, 5 - Select Multiply Final Adder. 6 - Select divide test points.

## GENERAL NETWORK

As shown in ALN 1.0, the general network consists of Branch Condition Control, Endcase Control, Exponent Arithmetic Network, Normalize Encoder, Shifter Ranks 1 and 2, Shift Count Generator, Large Adder, and ALN Output Mux.

The general network adds and subtracts integers and floating-point coefficients, and performs exponent arithmetic associated with all floating-point operations. Additionally it does boolean, floating-point normalize, shift, and conditional branch instruction test functions. The general network contains hardware to detect floating-point endcase conditions.

The general network has no holding registers and produces results in 64-ns major cycle intervals. If an intermediate result is produced, it must return to a Register File scratch register location in OPI. It then may pass through ALN again under the control of a second ALN Functional Unit Micrand or remain in the Register File while ALN tabulates the rest of the result. The 64-bit Functional Unit Micrand controls general network operation after entering the General Network Control/Micrand Register.

The general network receives C Operand integers directly from OPI for entry into the Shift Network Multiplexer, the Adder Multiplexer, or the Normalize Encoder. The B Operand integer enters the Shift Network or Adder Multiplexer by way of the Multiply/Divide Output Multiplexer when that hardware is not otherwise in use.

The 48 low-order bits of the B and C operands typically enter the Adder Multiplexer and shift network as coefficient values during floating-point add or subtract operations.

The Shift Count Generator ascertains which operand is larger by comparing exponents and controls selection of coefficients in the Adder and Shift Network multiplexers. The Shift Count Generator also produces a shift count to align them. When the C Operand exponent is larger than the B Operand exponent, the C Operand enters the Adder Multiplexer. The 48-bit B Operand coefficient enters the shift network for alignment with the C Operand coefficient before addition or subtraction occurs. The opposite selections occur if the B Operand exponent is larger. The coefficient result joins the larger exponent (selected in the Exponent Arithmetic Network) to become the ALN Result.

The general network performs C170 and C180 floating-point single- and double-precision addition and subtraction. In C170 mode, single-precision coefficient results are the high-order 48 bits of a 96-bit sum or difference. Double-precision results are the low-order 48 bits. In C180 mode single-precision coefficient results are a 48-bit sum or difference. Double-precision coefficient results are 96 bits of sum or difference. Refer to figure 4-12 for details of C180 96-bit double-precision addition.

The B and C Operand exponents enter the Exponent Arithmetic Network during all floating-point operations. B and C Operand exponents also enter Endcase Control to determine whether floating-point endcase conditions exist.

The general network performs exponent arithmetic for floating-point multiplication and division. The general network adds or subtracts exponents to create the new exponent value while the Multiply/Divide Network determines the coefficient.

The AC Shift Count supplies shift counts, bit string descriptors (C180 AC-AE instructions) or exponent fields for C170 pack (27) instructions. The AC Shift Count is an indirect input from OPI, passing first through address formation circuitry in AC. Because this path contains a holding register, values may be retained for subsequent ALN operations. ALN has no means to store them.

The general network delivers ALN results to OPI through the ALN Output Multiplexer.

#### SHIFT COUNT GENERATOR

The Shift Count Generator calculates shift counts required for use in the shift network. The Shift Count Generator is a 15-bit network consisting of an input multiplexer, unbiasing network, adder and output logic.

The unbiasing network removes C170 and C180 format biases from exponents. The adder employs C170 one's or C180 two's complement arithmetic. The output logic detects various shift faults and is capable of inverting adder results.

## SHIFT NETWORK

The shift network consists of the Shift Network Multiplexer and Shifter Ranks 1 and 2. The Shift Network Multiplexer produces a 128-bit output from various combinations of the B and C Operands, sign extension, or zero fill. The multiplexer contains a complement control, which may be used for magnitude form/signed operand conversions. The multiplexing logic permits selection of the coefficient with the smaller exponent during floating-point adds and subtracts. Operands may be right-shifted 64 places in the multiplexer.

Shifter Ranks 1 and 2 right shift the 128-bit multiplexer output from zero to 63 places, based on the shift count or the Significance Count. Left shifts end-around are accomplished by placing an operand and its duplicate end-to-end when forming the 128-bit Shift Network Multiplexer output. The two's complement of the shift count accomplishes the left shift by right-shifting the data from the high-order 64-bit field into the low-order 64-bit field.

## LARGE ADDER

The C Operand, the Exponent Network Result, the Normalize Count, and the B Operand enter the Adder Multiplexer. Various combinations of the four operands, sign extension, or zero fill produce a 96-bit Adder Multiplexer output. During floating-point add or subtract operations the multiplexer selects the B or C operand coefficient after the Shift Count Generator ascertains which coefficient has the larger exponent.

The 96-bit Adder Multiplexer output and the 96-bit Shift Network Output enter the Large Adder. The Large Adder is a 97-bit network (with end-round carry bit included) capable of 18-, 60- and 97-bit one's complement addition or subtraction. It also performs 32-bit and 64-bit two's complement addition or subtraction, and 64-bit and 32-bit AND, OR and EXCLUSIVE OR functions.

## NORMALIZE NETWORK

In floating-point configuration, an operand may be normalized by shifting the most significant bit of the coefficient to the high-order coefficient bit position. The exponent is reduced by the number of places the significant data is left-shifted.

The Significance Count is the number of leading zeros in the C Operand detected by the Normalize Encoder. When complemented, the Significance Count becomes the value the shift network uses to produce a right-shift equivalent to the normalizing left shift. Refer to Shift Network description of the left shift for explanation.

The Normalize Encoder also is used to encode a coefficient shift count to compensate for a detected coefficient overflow condition.

The Adjustment Adder modifies exponents with a constant field from the ALN Functional Unit Micrand. One example is the addition of 48<sub>10</sub> to the exponent during execution of a C170 single-precision floating-point multiply operation to account for the fact the single-precision result has a magnitude 48 bits larger than the entire 96-bit result.

## EXPONENT ARITHMETIC NETWORK

The Exponent Arithmetic Network performs 15-bit add, subtract, and multiplexing operations. An input multiplexer selects the B Operand, C Operand, Normalize Count, or AC Shift Count in pairs. An unbiased network removes C170 or C180 exponent format biases when required. After

one's or two's complement addition or subtraction occurs, the result enters control logic, which detects underflow or overflow conditions, complements the results, or inserts C170- or C180 format biases.

When floating-point addition or subtraction takes place, the Exponent Arithmetic Network selects the larger exponent value, based on a comparison in the Shift Count Generator. When floating-point multiply or divide operations occur, exponent addition or subtraction takes place.

The Normalize Count from the Adjustment Adder changes the C Operand exponent to reflect corresponding changes made to the normalized coefficient. During a C170 Pack (27) instruction the unpacked exponent contained in AC Shift Count Bits 0 through 11 is biased by the Exponent Arithmetic Network before it joins the coefficient in the ALN Output Multiplexer.

#### ALN OUTPUT MULTIPLEXER

The ALN Output Multiplexer is a 64-bit network which sends ALN Result to OPI. The multiplexer may select 64 bits from the shift network or the Large Adder. It also may select 48-bit coefficients from these sources and combine them with exponents from the Exponent Arithmetic Network or Endcase Control.

#### ENDCASE CONTROL

Endcase logic monitors the B and C Operand exponents during floating-point arithmetic. If out-of-range or nonstandard values are detected, the network generates the appropriate endcase result for the operation being performed. That result typically passes through the Normal/Endcase Exponent Selector in place of the arithmetic result. ICC examines ALN Exceptions to determine whether an interrupt routine must be initiated.

#### BRANCH CONDITION CONTROL

Branch Condition Control determines whether to generate ALN Unbranch based on Large Adder Zero Tests (various B and C Operand tests occurring in Endcase Control) or normalize tests for significant data. Instruction Fetch (IF) terminates its branched to instruction routine on receipt of ALN Unbranch. An ALN Functional Unit Micrand field specifies the condition on which the unbranching decision is based.

#### MULTIPLY/DIVIDE NETWORK

The Multiply/Divide Network (ALN 1.0) contains the hardware necessary to form the C180 integer product or C170, C180 floating-point product coefficient from the B (multiplicand) and C (multiplier) operands. The product may be represented as a 32- or 64-bit C180 signed integer. It also may be a 48-bit signed coefficient for C170 floating-point operations, or a 48-bit or 96-bit coefficient in magnitude form (absolute value accompanied by sign bit) for C180 floating-point operations.

The network performs divide operations in which the quotient of the B (dividend) and C (divisor) operands may be represented as a 32- or 64-bit signed integer (C180 mode only), or in one of the following floating-point coefficient formats.

The Multiply/Divide Network produces floating-point coefficients in the following formats:

- C170 double-precision product - low-order 48 bits of 96-bit result.
- C170 single-precision product - high-order 48 bits of 96-bit result.
- C170 single-precision quotient - 48-bit result selected.
- C180 single-precision quotient - 48-bit result selected.
- C180 double-precision quotient - 96-bit result selected.
- C180 single-precision product - high-order 48 bits of 96-bit result.
- C180 double-precision product - high-order 96 bits of 192-bit result.

The C170 Population Counter counts all logical one bits in a 60-bit operand during a C170 Population Count (47) instruction.

The Multiply/Divide Network assists BDP when BDP converts Binary Coded Decimal (BCD) digits into binary form. BDP converts the digits from BCD to binary form one pair at a time before sending the data to ALN. The multiply network multiplies the running total of BCD digits by  $100_{10}$  before adding two new binary form BCD digits to that total.

The B Operand uses a path through the Multiply/Divide Output Multiplexer to enter the general network for various operations. The C Operand enters the C Register on its way to the Multiply/Divide Output Multiplexer and ALN Output Multiplexer during an implicit copy operation. Implicit copy operations transfer the C Operand from one Register File location to another in the absence of an ALN Functional Unit Micrand.

Fields within the ALN Functional Unit Micrand control the operation of the Multiply/Divide Network on major cycle boundaries. Soft Control governs operations on minor cycle boundaries. The ALN Functional Unit Micrand supplies five-bit partial addresses for Soft Control routines as well as 15 bits for direct control of multiply or divide operations.

## MULTIPLICATION

The Multiply/Divide Network includes a 12-bit by 64-bit multiplication circuit which creates a partial product in magnitude form every 16 ns. The B Register contains the multiplicand. The C Register serves as a holding register for the multiplier (C Operand) and C170 double-precision product coefficient, C170 integer product or C180 integer product. During each iteration the least significant 12 bits of the multiplier enter the Product Networks A, B from the C Register. The remaining multiplier bits are right-shifted 12 places in the C Register. The 12 least significant bits of the multiply partial product produced in the Multiply Final Adder enter the vacated high-order bit positions in the C Register. This process repeats each iteration.

During C180 integer multiplication, 32- or 64-bit products are output from the C Register to OPI through the general network. C170 double-precision floating-point product coefficients likewise proceed from the C Register to the general network. The C170 single-precision and C180 single-precision product coefficients are available to the general network from the Multiply Final Adder.

The Exponent Arithmetic Network in the general network adds floating-point exponent values during multiply operations. The general network also converts integer products and C170 floating-point product coefficients from magnitude form to signed operands.

Refer to figure 4-1 for simplified view of multiplication data paths.

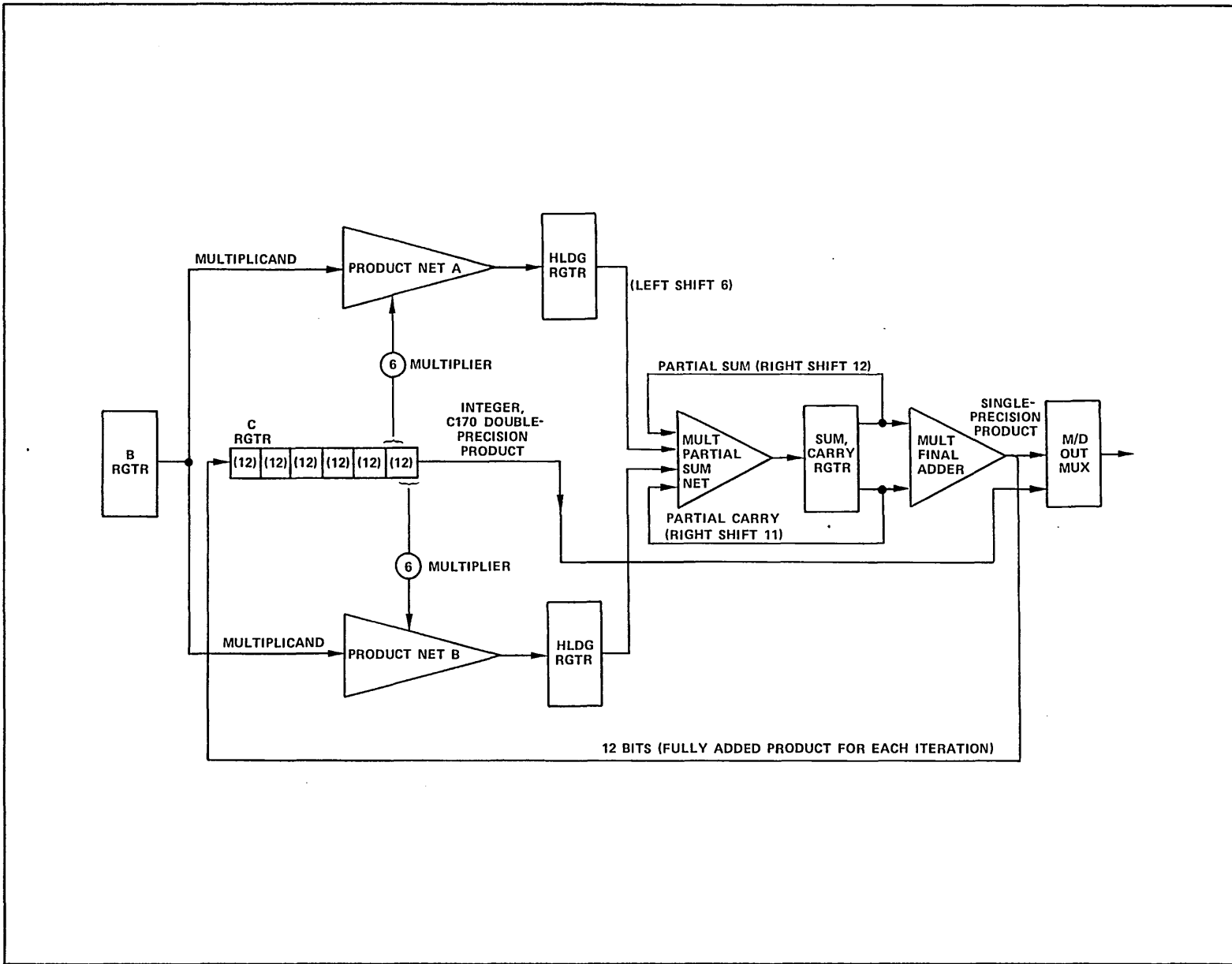


Figure 4-1. Multiplication Network

### B,C Registers

The B Register supplies the multiplicand to Product Networks A, B. A multiplexer at the register input selects the B Operand from OPI or BDP convert-to-binary intermediate results from the Multiply Final Adder. Because the B Register loads new data every 16 ns, the multiplicand must be available from OPI to the B Register during the entire multiplication.

The C Register serves as the multiplier register. It also accumulates product results 12 bits at a time. Typically, a shifted multiplier value and new partial product enter the C Register at 16-ns intervals during the multiplication.

The multiply algorithm requires that 13 multiplier bits from the operand be input to Product Networks A, B each minor cycle (12 new bits and a duplicate bit from the previous iteration).

### Product Networks A,B

Product Networks A, B multiply 12 bits of multiplier times the multiplicand from the B Register each minor cycle. The multiplication is based on Booth's Algorithm. Refer to figure 4-2 for a demonstration of the algorithm as it is applied in the product networks.



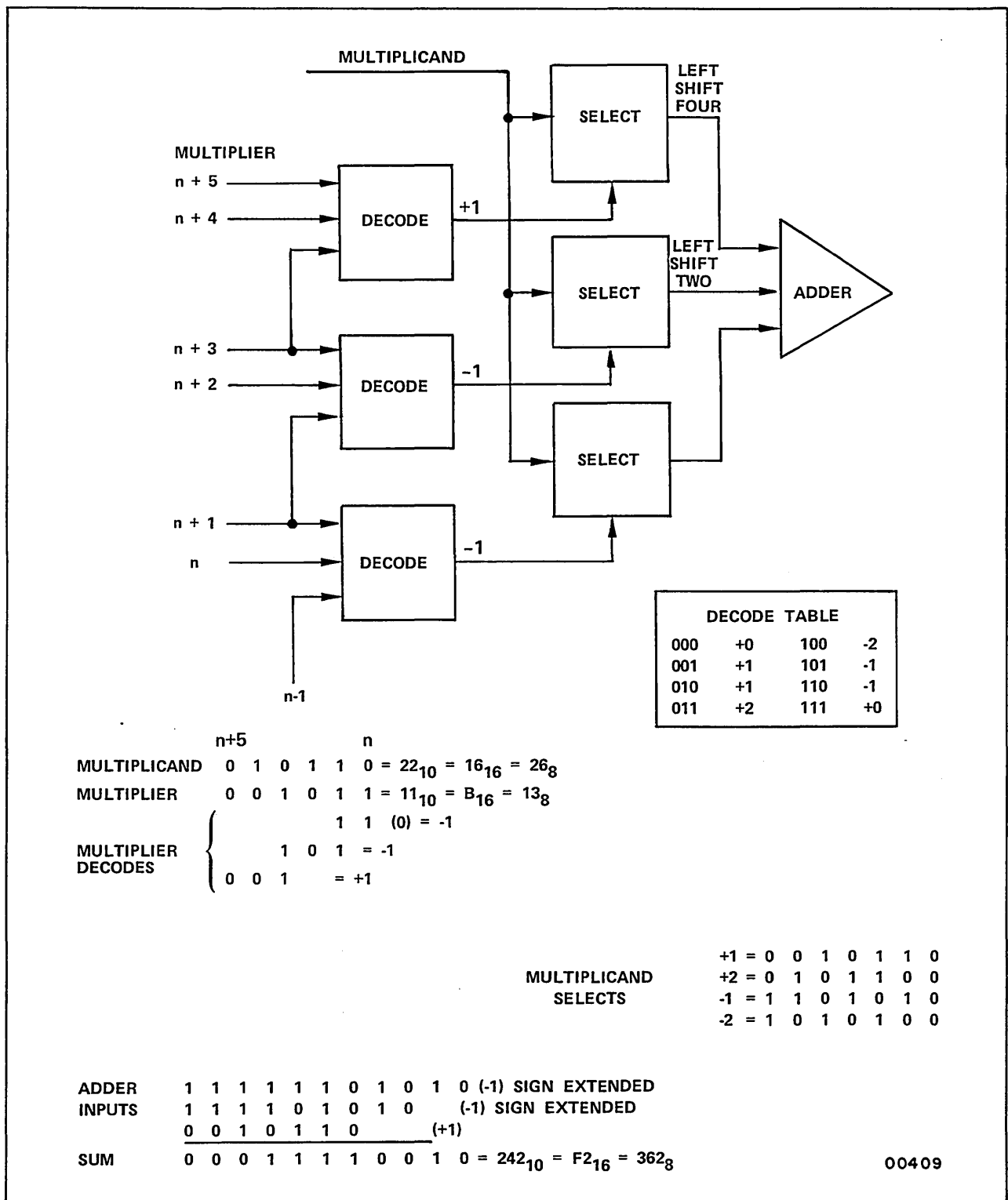


Figure 4-2. Demonstration of Booth's Algorithm

#### NOTE

Booth's Algorithm relies on the fact that the value of a given bit string (an uninterrupted sequence of logic ones) can be represented by the algebraic expression  $2^{m+1}-2^n$ .  $m$  represents the bit position of the high-order bit of the string and  $n$  is the bit position of the low-order bit of the string.

The 12 bits of the Product Network A, B multiplier (ALN 1.0) are subdivided into six pairs. Each pair is accompanied by a duplicate of the high-order bit of the next lower pair. For the lowest-order pair, this bit is the high-order bit of the next lower multiplier. Bit duplicating aids in determining where bit strings begin and end.

The algorithm assigns a value to each of these three-bit groups based on the algebraic expression. The assigned values appear in the decode table, figure 4-2. Each multiplier group is given a value of 0, +1, -1, +2, or -2.

Multiplication is then accomplished by entering zeros, passing a given multiplicand unshifted, or left-shifting it one place. One's complementing of the multiplicand occurs if the multiplier and the multiplicand have unlike signs or the decoded two-bit multiplier value is negative.

In each product network, three partial products are produced, one left-shifted four places, another two places. The third remains unshifted. These manipulations account for the difference in magnitude of the three two-bit multipliers in each product network. The three partial products are added, resulting in network partial sum and carry bits.

The output of Product Network A is left-shifted six places with respect to the output of Product Network B to account for the difference in magnitude of the two six-bit multipliers.

The multiplier values shown in the decode table (figure 4-2) are derived as follows. Consider a two-bit multiplier in the following form,  $2^1, 2^0$ , Duplicate Bit. Recall the expression  $2^{m+1} - 2^n$ . If  $2^0$  is a zero and the Duplicate Bit is a one, the upper end of a bit string has been located.  $2^0$  is  $2^{m+1}$  of the algebraic expression. If  $2^1$  in this same multiplier is a one, it represents  $2^n$  of the next bit string in the multiplier.

When this bit pair is evaluated in isolation,  $2^1$  has a weight of  $-2$  ( $2^n$  is always negative) and  $2^0$ , a weight of 1. (Even though  $2^0$  is a zero, Booth's Algorithm assigns it a value of 1 because it is  $2^{m+1}$ .) Addition produces  $-1$  from these two quantities, which the decode table indicates is the value assigned to binary 5.

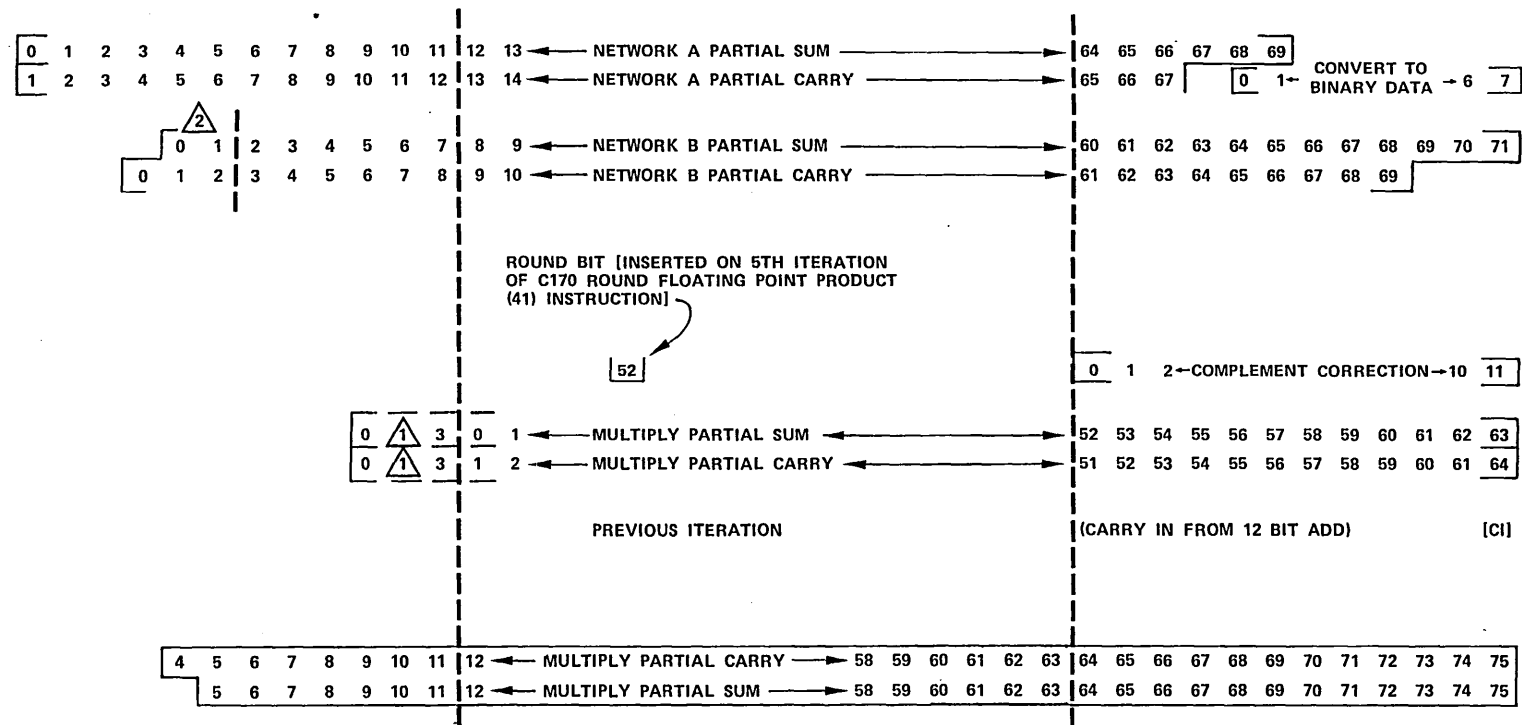
The key to understanding Booth's Algorithm in this context is that the pairing of multiplier bits in most cases prohibits calculating an algebraic expression's value directly. In the example shown, two halves of unrelated algebraic expressions are combined arithmetically. By itself, the calculation is incorrect. But the consequence of applying this method to an entire multiplier is the correctly calculated value of all bit strings appearing in it. The advantage implicit in the product network design is that it reduces the multiplier's size by 1/2. A single operation is performed for every two bits of multiplier.

#### Multiply Partial Summing Network

Product Networks A, B contain partial adders to produce partial sums and carries, which are then fed to the Multiply Partial Summing Network. The Multiply Partial Summing Network adds seven separate inputs to produce a Multiply Partial Sum bit and a Multiply Partial Carry bit for each bit position. Seventy-two Multiply Partial Sum bits and 71 partial carry bits enter the Multiply Final Adder.

Four of the inputs to the network are from Product Networks A, B. In addition, partial sums and carries from the previous multiply iteration enter the network right-shifted 12 places to account for the difference in magnitude. The seventh input is an adjustment of the multiplicand, which may be one's complemented in a product network. The partial summing network performs two's complement addition.

Refer to figure 4-3 for a illustration of the multiplier partial summing network inputs and outputs.



NOTES: 1 EXTRA PRECISION BITS REQUIRED FOR 64-BIT UNSIGNED MULTIPLICATION.  
2 SIGN EXTENSION BITS 0, 1 ARE SIGN EXTENSION.

00407

Figure 4-3. Multiply Partial Summing Network Inputs, Outputs

#### BDP Decimal-to-Binary Conversion

When converting BCD values to binary form, BDP requires assistance from the Multiply/Divide Network. Two BCD digits are converted to a binary representation in BDP and sent to ALN. When the next two converted digits arrive, they are added to the previous value which has been multiplied by  $100_{10}$ . Multiplication is necessary because the previous value has magnitude two decimal places greater than the incoming digits. A new running total returns to the B Register after addition occurs. The C Register contains the constant multiplier ( $100_{10}$ ).

Multiplication of the running total by  $100_{10}$  occurs in the multiplication network. Converted BCD digits enter the Multiply Partial Summing Network to be added in the final stages of each multiplication.

Refer to figure 4-4 for an example of decimal-to-binary operations in ALN.

ALGORITHM =  $\{ [ (0 \times 100_{10} + A) 100_{10} + B ] 100_{10} + C \}$

| CHARACTER 1 | CHARACTER 2 | CHARACTER 3-9 |

BCD CHARACTER STRING = 46 (=2E<sub>16</sub>),      60 (=3C<sub>16</sub>),      X,X,X, —————> X

#### FIRST CHARACTER (PASS)

MULTIPLY 0 X 100<sub>10</sub>(64<sub>16</sub>) + FIRST CHARACTER  
AND STORE IN B REGISTER

0 ————— 0 (B RGTR)  
0 ————— 001100100 (C RGTR)  
0 ————— 0  
————— 00101110 (CONVERT BYTE)  
0 ————— 000101110 B RGTR

(8 BITS OF SIGNIFICANCE)

#### SECOND CHARACTER

MULTIPLY FIRST CHARACTER X 100<sub>10</sub>  
+SECOND CHARACTER  
AND STORE IN B REGISTER.

0 ————— 0101110 (B RGTR) (46<sub>10</sub>)  
X0 ————— 01100100 (C RGTR) (100<sub>10</sub>)  
—————  
(0) 0 ————— 0  
(+1) 0 ————— 010111000  
(-2) 1 ——— 110100100  
(+2) 0 ——— 01011100

(8 BITS OF SIGNIFICANCE X 7 BIT MULT. =  
15 BITS OF SIGNIFICANCE)

0 ——— 01000111111000 (4600<sub>10</sub>)  
————— 00111100 (CONVERT BYTE) (60<sub>10</sub>)  
0-0001 0010 0011 0100 (B RGTR) (4660<sub>10</sub>)  
    1      2      3      4 16

#### THIRD THROUGH NINTH CHARACTER

EACH CHARACTER WILL ADD 7 BITS OF SIGNIFICANCE  
TO THE RUNNING TOTAL. AFTER 9 CHARACTERS THE  
64 BIT TOTAL MUST BE STORED IN A SCRATCH  
REGISTER. CHARACTER STRINGS LONGER THAN  
9 BYTES AND LESS THAN OR EQUAL TO 19 BYTES  
REQUIRE 2 SCRATCH REGISTERS

Figure 4-4. Decimal-to-Binary Conversion

### Multiply Final Adder

The Multiply Final Adder combines the partial sums and carries from the Multiply Partial Summing Network. The low-order 12 bits of the Multiply Result are the partial product of a given iteration. The partial product returns to the C Register at the same time the right-shifted multiplier returns to the C Register.

During BDP conversion or at the end of single-precision floating-point multiplication, the result is available to the Multiply/Divide Output Multiplexer from the Multiply Final Adder.

### Multiply/Divide Output Multiplexer

The Multiply/Divide Output Multiplexer can select multiply results from either the C Register or the Multiply Final Adder.

## DIVISION

The Multiply/Divide Network performs C180 integer and C170, C180 floating-point divide operations. All results are expressed in magnitude form and are converted to proper signed values by the general network when required.

The Divide Network consists of a quotient register, and add/subtract circuitry, which includes a divisor and dividend register. The quotient register provides a path for a 48-bit B Operand coefficient to enter the dividend register during a floating-point operation. During a C180 integer divide operation the quotient register holds the dividend and sends the most significant dividend bit to the dividend register each iteration. During both operations dividends are converted to magnitude form before the actual calculation begins.

The C Operand enters the divisor register. The contents of the divisor and dividend registers enter the adder/subtractor each minor cycle to produce a quotient bit. The quotient bit enters the quotient register as the least significant bit. Each iteration produces an upward shift of the quotient in the quotient register. By the time the divide operation ends, the entire quotient has entered the quotient register.

Refer to figure 4-5 for a simplified explanation of the restoring divide algorithm used by the Divide Network. Refer to figure 4-6 for explanation of how a quotient coefficient is calculated during floating-point division.

During the first iteration of integer division the contents of the divisor register (figure 4-5) are subtracted from zero to test whether the divisor is equal to zero (divide fault). The carry produced in the subtractor (a zero if divide fault does not occur) determines whether addition or subtraction occurs in the next minor cycle. A logical zero carry out from any iteration indicates the divisor is larger than the portion of the dividend it has been divided into (or, in the case of the divide fault test, larger than zero).

In any divide operation, the dividend is enlarged one place if, in the preceding step, the divisor is larger than the dividend. If it is larger, the computer's divide algorithm recalculates the original dividend from the remainder and the divisor. It then left-shifts the sum one place, introducing a new dividend bit. The expression for the next divide attempt is  $2(\text{remainder} + \text{divisor}) - \text{divisor}$ . The Divide Network simplifies this computation by left-shifting the remainder one place and using the zero carry out to force addition of the shifted remainder and the divisor.

Algebraically, the two calculations are equivalent, that is,  $2(\text{remainder} + \text{divisor}) - \text{divisor} = 2 \text{ remainder} + \text{divisor}$ .  $2 \text{ remainder} + \text{divisor}$  represents the result of a divide step following a divide step that produced a zero quotient bit.

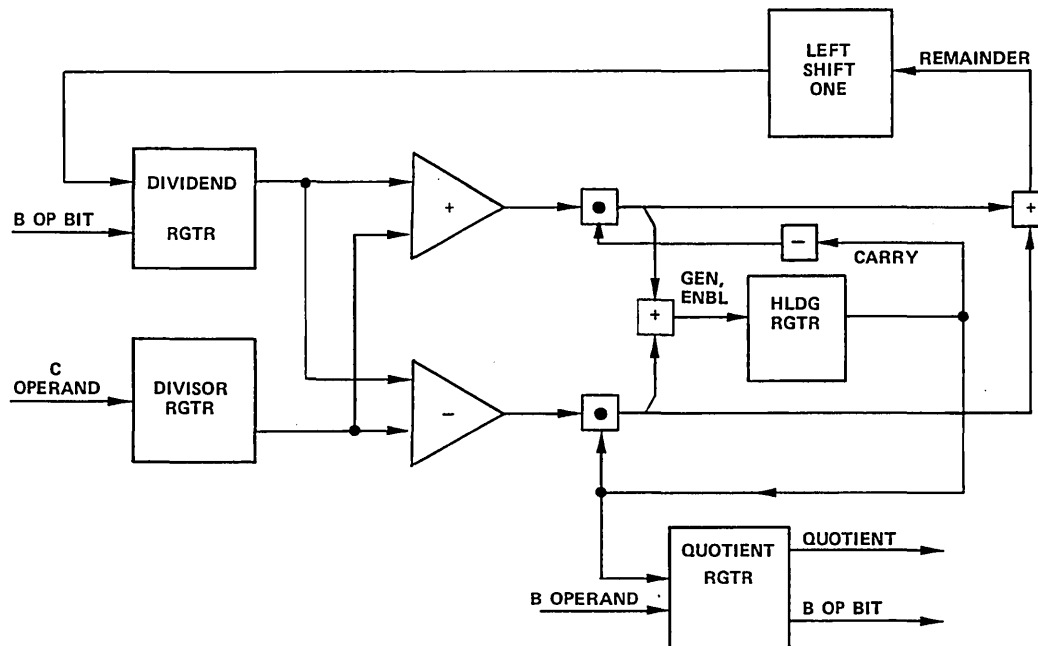
If the carry produced by  $2 \text{ remainder} + \text{divisor}$  is also a zero, the process repeats, with a second zero bit being placed in the quotient register. If addition in any following iteration results in a logical one carry out, a one bit is placed in the quotient register and the divisor is subtracted from the left-shifted remainder.

During a floating-point divide operation (figure 4-6) the first iteration subtracts 2 times Divisor from the Dividend to ascertain whether the Dividend is  $\geq 2$  times Divisor. If the Dividend meets this condition, the carry bit (logical one) indicates a divide fault after entering the quotient register. The balance of the floating-point divide operation parallels integer division, with a succession of adds or subtracts of the divisor from the shifted remainder. It differs from integer division in that the dividend is available in the Dividend Register from the beginning.

During floating-point division the exponent of the divisor is subtracted from the exponent of the dividend to produce the exponent of the quotient in the Exponent Arithmetic Network.

Once integer/floating-point division is complete, the quotient/coefficient enters the general network by way of the C Register and the Multiply/Divide Output Multiplexer.





### INTEGER DIVIDE EXAMPLE

	ADDER/SUBTRACTER	QUOTIENT RGTR
6 BIT DIVIDEND	0 0 0 0 0 0	0 1 1 1 0 = $30_{10} = 1E_{16} = 36_8$
6 BIT DIVISOR	$\triangle 2$ - 0 0 0 1 1 0	= $6_{10} = 6_{16} = 6_8$
1ST ITERATION RESULT	$\triangle 1$ 0 1 1 0 1 0	(DIVIDE FAULT TEST)
LEFT SHIFT ONE	1 1 0 1 0 0	1 1 1 1 0 0
	+ 0 0 0 1 1 0	
2ND ITERATION RESULT	0 1 1 0 1 0	
LEFT SHIFT ONE	1 1 0 1 0 1	1 1 1 0 0 0
	+ 0 0 0 1 1 0	
3RD ITERATION RESULT	0 1 1 0 1 1	
LEFT SHIFT ONE	1 1 0 1 1 1	1 1 0 0 0 0
	+ 0 0 0 1 1 0	
4TH ITERATION RESULT	0 1 1 1 0 1	
LEFT SHIFT ONE	1 1 1 0 1 1	1 0 0 0 0 0
	+ 0 0 0 1 1 0	
5TH ITERATION RESULT	1 0 0 0 0 1	
LEFT SHIFT ONE	0 0 0 0 1 1	0 0 0 0 0 1
	$\triangle 2$ - 0 0 0 1 1 0	
6TH ITERATION RESULT	0 1 1 1 0 1	
LEFT SHIFT ONE	1 1 1 0 1 0	0 0 0 0 1 0
	+ 0 0 0 1 1 0	
7TH ITERATION RESULT	1 0 0 0 0 0	
LEFT SHIFT ONE	0 0 0 0 0 0	0 0 0 1 0 1 = $5_{10}$

#### NOTES:

- $\triangle 1$  A LOGICAL 1 CARRY INDICATES ADDER/SUBTRACTER RESULT IS POSITIVE.
- $\triangle 2$  TWOS COMPLEMENT OF DIVISOR IS ADDED TO DIVIDEND TO PERFORM SUBTRACTION.

Figure 4-5. Integer Division

6 BIT DIVIDEND = 011 110 ( $36_8$ )  
 6 BIT DIVISOR = 000 110 ( $6_8$ )

PACKED AND NORMALIZED DIVIDEND = 111 100  $\times 2^{-1}$   
 PACKED AND NORMALIZED DIVISOR = 110 000  $\times 2^{-3}$   
 $2^{+2}$  = EXPONENT DIFFERENCE

1. FIRST ITERATION. TEST FOR DIVIDE FAULT, DIVIDEND MINUS 2 TIMES THE DIVISOR. IF SUBTRACTOR CARRY EQUALS 0, INSERT A 0 IN THE QUOTIENT, AND EXAMINE ADDER RESULTS NEXT ITERATION. IF SUBTRACTOR CARRY EQUALS 1, (DIVIDEND  $\geq$  2 TIMES THE DIVISOR), INSERT A 1 IN THE QUOTIENT AND EXAMINE SUBTRACTION RESULTS NEXT ITERATION. ITERATION RESULTS ARE LEFT SHIFTED BACK INTO THE ADDER AND SUBTRACTOR (DIVIDEND INPUT) FOR THE NEXT ITERATION.

①  $\triangle$  011110 (DIVIDEND RIGHT SHIFTED 1)  
 $\underline{-110000}$   
 0101110 = CARRY = 0, INSERT 0 IN QUOTIENT, EXAMINE ADDER RESULTS  
 NEXT ITERATION. QUOTIENT = 0X.XXXXXX

LEFT SHIFT .1011100  
 ONE

2. SECOND THROUGH FIFTH ITERATION. EXAMINE ADDER OR SUBTRACTOR CARRY FOR THIS ITERATION BASED ON CARRY OF PREVIOUS ITERATION (0 = ADDER, 1 = SUBTRACTOR). ITERATION CARRY DETERMINES QUOTIENT BIT.

②  $\begin{array}{r} 1011100 \\ +0110000 \\ \hline \end{array}$   
 1 0001100 QUOTIENT = 01.XXXX

③ LEFT SHIFT ONE  $\triangle$  0011000  
 $\underline{-0110000}$   
 0 1101000 QUOTIENT = 01.0XXX

④ LEFT SHIFT ONE 1010000  
 $\underline{+0110000}$   
 1 0000000 QUOTIENT = 01.01XX

⑤ LEFT SHIFT ONE  $\triangle$  0000000  
 $\underline{-0110000}$   
 0 1010000 QUOTIENT = 01.010X

RESULT = 01.010000 (RESULT FORMAT = XX.XXXXXX)

↑  
 DIVIDE FAULT = 0

NOTES:

UNPACKED RESULT = 101.0 -0 =  $5_8$

$\triangle$  TWOS COMPLEMENT OF DIVISOR IS ADDED TO  
 DIVIDEND TO PERFORM SUBTRACTION.

Figure 4-6. Floating-Point Division

## C170 POPULATION COUNTER

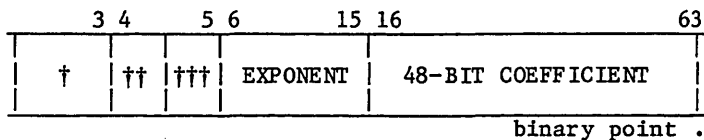
During a C170 Population Count (47) instruction the B Operand enters the quotient register in the Divide Network. The contents of the quotient register are left-shifted one place each minor cycle. The C170 Population Counter increments each time a logical one enters the Quotient Bit 4 position.

## C170/C180 DIFFERENCES

Numerous differences exist in the way ALN executes C170 and C180 product set instructions. These differences result from the existence of different floating-point operand formats, different operand sizes, differing instruction requirements, and C170 mode's reliance on one's complement arithmetic versus C180 mode's use of two's complement arithmetic.

## FLOATING-POINT OPERATIONS

The CYBER 170 state single- or double-precision floating-point operand is shown in the following format.



† Coefficient sign extension

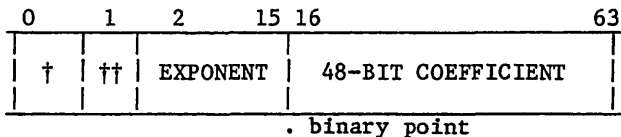
†† Coefficient sign

††† Exponent bias

The 48-bit coefficient is in signed form, that is, negative values are expressed in one's complement form. The exponent bias is the complemented exponent sign. The exponent itself is in signed form. If the coefficient sign is negative, the exponent and exponent bias are complemented. The coefficient is an integer with the binary point to the right of the bit 63 position.

Single-precision C170 instructions use the upper 48 bits of the coefficient; double-precision instructions use the lower 48 bits.

The C180 state single-precision floating-point operand is shown in the following format.

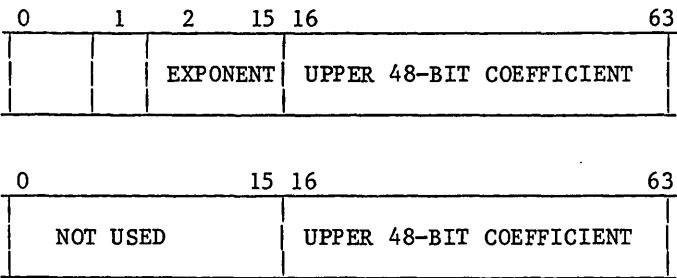


† Coefficient sign

†† Exponent bias

The coefficient is in magnitude form (value without regard for sign, expressed in true form). The exponent bias is the complemented exponent sign. The exponent itself is in signed (two's complement) form. The coefficient is a fraction with the binary point located between bit positions 15 and 16.

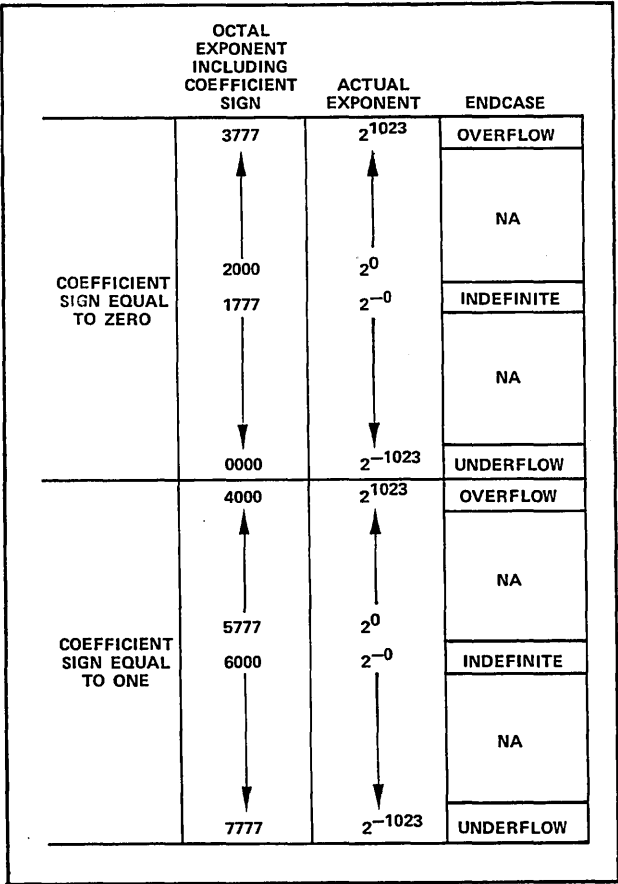
The C180 state double-precision floating-point operand appears in the following format.



Coefficient sign  
Exponent bias

Floating-Point Exponent Ranges

Floating-point exponent ranges, expressed in octal notation for C170 floating-point operands, are shown in figure 4-7. Floating-point exponent ranges, expressed in hexadecimal notation for C180 floating-point operands, are shown figure 4-8.



00411

Figure 4-7. C170 Exponents

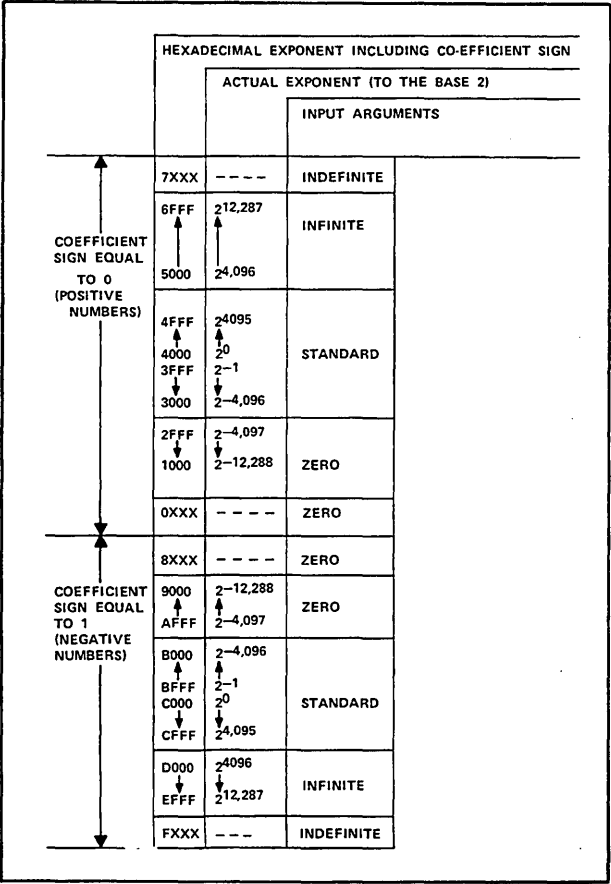


Figure 4-8. C180 Exponents

## Endcase Conditions

Figures 4-9 and 4-10 indicate the endcase conditions that result from operand inputs of various types in C170 and C180 modes.

Xj PLUS Xk (30, 32, 34 INSTRUCTIONS)						Xj MINUS Xk (31, 33, 35 INSTRUCTIONS)					
		Xk						Xk			
		W	+∞	-∞	± IND			W	+∞	-∞	± IND
Xj	W		+∞	-∞	IND	Xj	W		-∞	+∞	IND
	+∞	+∞	+∞	IND	IND		+∞	+∞	IND	+∞	IND
	-∞	-∞	IND	-∞	IND		-∞	-∞	-∞	IND	IND
	± IND	IND	IND	IND	IND		± IND	IND	IND	IND	IND

W Any word except ± ∞ and ± IND

W Any word except ± ∞ and ± IND

Xj MULTIPLIED BY Xk (40, 41, 42 INSTRUCTIONS)								
		Xk						
		+N	-N	+0	-0	+∞	-∞	± IND
Xj	+N			0	0	+∞	-∞	IND
	-N			0	0	-∞	+∞	IND
	+0	0	0	Integer <sup>†</sup> multiply		IND	IND	IND
	-0	0	0			IND	IND	IND
	+∞	+∞	-∞	IND	IND	+∞	-∞	IND
	-∞	-∞	+∞	IND	IND	-∞	+∞	IND
	± IND	IND	IND	IND	IND	IND	IND	IND

† If both operands used in the integer multiply are normalized, an underflow results.

N Any word except ± ∞, ± IND, and ± 0

Xj DIVIDED BY Xk (44, 45 INSTRUCTIONS)								
		Xk						
		+N	-N	+0	-0	+∞	-∞	± IND
Xj	+N			+∞	-∞	0	0	IND
	-N			-∞	+∞	0	0	IND
	+0	0	0	IND	IND	0	0	IND
	-0	0	0	IND	IND	0	0	IND
	+∞	+∞	-∞	+∞	-∞	IND	IND	IND
	-∞	-∞	+∞	-∞	+∞	IND	IND	IND
	± IND	IND	IND	IND	IND	IND	IND	IND

00228

N Any word except ± ∞, ± IND, and ± 0

00228

Figure 4-9. C170 Floating-Point Endcase Results

## ADD, SUBTRACT OPERANDS

		Xj			
		W	+x	-x	± IND
Xk	W	S	+x	-x	IND
	+x	+x	+x	IND	IND
	-x	-x	IND	-x	IND
	± IND	IND	IND	IND	IND

$$(Xk) + (Xj) \text{ AND } (Xk, Xk + 1) + (Xj, Xj + 1)$$

		Xj			
		W	+x	-x	± IND
Xk	W	D	-x	+x	IND
	+x	+x	IND	+x	IND
	-x	-x	-x	IND	IND
	± IND	IND	IND	IND	IND

$$(Xk) - (Xj) \text{ AND } (Xk, Xk + 1) - (Xj, Xj + 1)$$

## ADD, SUBTRACT RESULT TEST

1. COEFFICIENT OVERFLOW RIGHT SHIFT COEFFICIENT 1 PLACE AND INCREASE EXPONENT BY 1
2. EXPONENT OVERFLOW
  - A. C170 MODE OR C180 MODE · UMR BIT; FORCE INFINITE EXPONENT AND CLEAR COEFFICIENT.
  - B. C180 MODE · UMR BIT; SEND RESULT EXPONENT AND COEFFICIENT OUT.
3. LOSS OF SIGNIFICANCE (C180 ONLY)
  - A. UMR BIT; FORCE RESULT EQUAL TO F.P. ZERO.
  - B. UMR BIT; SEND RESULT EXPONENT AND COEFFICIENT.
4. EXPONENT UNDERFLOW (C180 COEFFICIENT NORMALIZATION)
  - A. UMR BIT; FORCE RESULT EQUAL TO F.P. ZERO.
  - B. UMR BIT; SEND RESULT EXPONENT AND COEFFICIENT OUT

## MULTIPLY OPERANDS

		Xj					
		+N	-N	+0	-0	+x	-x ± IND
Xk	+N	+P	-P	0	0	+x	-x IND
	-N	-P	+P	0	0	-x	+x IND
	+0	0	0	0	0	IND	IND IND
	-0	0	0	0	0	IND	IND IND
	+x	+x	-x	IND	IND	+x	-x IND
	-x	-x	+x	IND	IND	-x	+x IND
	± IND	IND	IND	IND	IND	IND	IND IND

$$(Xk) \cdot (Xj) \text{ AND } (Xk, Xk + 1) \cdot (Xj, Xj + 1)$$

## MULTIPLY RESULT TEST

1. COEFFICIENT NORMALIZE IF INPUT OPERANDS ARE NORMALIZED AND RESULT IS NOT NORMALIZED, LEFT SHIFT COEFFICIENT 1 PLACE AND DECREASE EXPONENT BY 1.
2. EXPONENT OVERFLOW
  - A. C170 MODE OR C180 MODE · UMR BIT; FORCE INFINITE EXPONENT AND CLEAR COEFFICIENT.
  - B. C180 MODE · UMR BIT; SEND RESULT EXPONENT AND COEFFICIENT.
3. EXPONENT UNDERFLOW
  - A. C170 MODE OR C180 MODE · UMR BIT; FORCE UNDERFLOW EXPONENT AND CLEAR COEFFICIENT.
  - B. C180 MODE · UMR BIT; SEND RESULT EXPONENT AND COEFFICIENT.

## DIVIDE OPERANDS

		Xj					
		+N	-N	+0	-0	+x	-x ± IND
Xk	+N	+Q	-Q			0	0 IND
	-N	-Q	+Q			0	0 IND
	+0	0	0	Xk	WITH	0	0 IND
	-0	0	0			0	0 IND
	+x	+x	-x	DIVIDE FAULT		IND	IND IND
	-x	-x	+x			IND	IND IND
	± IND	IND	IND			IND	IND IND

$$(Xk)/(Xj) \text{ AND } (Xk, Xk + 1)/(Xj, Xj + 1)$$

## DIVIDE RESULT TEST

- DIVIDE FAULT (Xk COEFFICIENT  $\geq 2$  Xj COEFFICIENT OR Xj = 0)
- A. C170 MODE; FORCE INDEFINITE EXPONENT AND CLEAR COEFFICIENT.
  - B. C180 MODE · UMR BIT; FORCE INDEFINITE EXPONENT AND CLEAR COEFFICIENT.
  - C. C180 MODE · UMR BIT; FORCE RESULT EQUAL TO Xk OPERAND.
- COEFFICIENT OVERFLOW RIGHT SHIFT COEFFICIENT 1 PLACE AND INCREASE EXPONENT BY 1.
- EXPONENT OVERFLOW  
SAME AS MULTIPLY
- EXPONENT UNDERFLOW  
SAME AS MULTIPLY

## KEY TO SYMBOLS:

W	ANY WORD EXCEPT x, IND
N	ANY WORD EXCEPT x, IND, OR 0
S	ALGEBRAIC SUM PER INSTRUCTION DESCRIPTIONS
D	ALGEBRAIC DIFFERENCE PER INSTRUCTION DESCRIPTIONS
P	PRODUCT PER INSTRUCTION DESCRIPTIONS
Q	QUOTIENT PER INSTRUCTION DESCRIPTIONS
x	INFINITE
IND	INDEFINITE
0	ZERO

Figure 4-10. C180 Floating-Point Endcase Results

### Integer/Floating-Point Conversions

In C170 mode, pack and unpack (27,26) instructions convert coefficients and exponents contained in separate registers to floating-point operands and vice versa.

In C180 mode, Convert from Integer to Floating-Point (3A) and Convert from Floating-Point to Integer (3B) instructions produce 64-bit floating-point operands from 64-bit integers and vice versa.

### Normalization and Rounding

In C170 mode, a Normalize (24) instruction normalizes the floating-point operand read from the Register File's  $X_k$  Register and places it in the  $X_i$  Register. At the end of the instruction the B Register in the Register File contains the left-shift count required to normalize the coefficient. A normalized floating-point operand is defined as one which has the most significant bit in the leftmost coefficient bit position (bit 16).

In a C180 mode instruction normalization of floating-point results is performed by the microcode as a standard step. A normalized floating-point operand is defined as one which has a logical one in the most significant coefficient bit position.

The rounding operation adds a bit in the position immediately below the least significant bit position in the floating-point coefficient. This feature exists only in C170 mode.

### OPERANDS

In C170 mode, ALN typically performs arithmetic operations on 18- or 60-bit operands. In C180 mode, the operands usually contain 32 or 64 bits. C170 18-bit operands are zero-filled in the remaining 46-bit positions of a 64-bit word. C170 60-bit operands are sign-extended in the four high-order bit positions of a 64-bit word. C180 32-bit operands may be sign-extended or zero-filled in the remaining bit positions.

### CONDITIONAL BRANCH INSTRUCTIONS

Conditional branch instructions are evaluated by Branch Condition Control in the general network. If the specified condition is not met, ALN issues ALN Unbranch to the Instruction Unit, indicating the branched to routine should not be executed.

In C170 mode, conditional branch instructions typically examine the sign or value contained in a 60-bit X Register to determine if a branch routine is to be executed. In the case of four C170 conditional branch instructions, comparison of the contents of two 16-bit B Registers forms the basis of the branch/unbranch decision.

Conditional branch instructions in C180 mode most often are based on ALN's comparison of two operands from the Register File. The comparisons occur between floating-point operands and between 64- or 32-bit (half-word) integers.

In either mode, branch testing can be based on the presence or absence of designated floating-point endcase conditions.

## INSTRUCTION SEQUENCES

The following five instruction sequences describe the typical use of ALN hardware. The instructions are C170 Right Shift (21), C180 Integer Add (24), C180 Single-Precision Floating-Point Sum (30), C170 Single-Precision Floating-Point Product (40), and C180 Integer Quotient (27) instructions. In addition, the sequence of events associated with an implicit copy operation is described. References are to Level 1 ALN diagram in the instruction descriptions and to Level 3 diagrams within the sequences. The operations of ALN under Soft Control Data Output and the ALN Functional Unit Micrand are described in detail. The roles of the Register File in OPI and the Instruction Unit are considered only briefly. The theory of their operation appears in section 3.

### C170 RIGHT SHIFT (21) INSTRUCTION

The instruction right-shifts a 60-bit operand from an X register specified by the instruction's i field. The j and k fields (together forming a six-bit positive integer) determine the amount of the shift. The shift is end-off, with bits shifted beyond the least significant bit position being discarded. The 60-bit result returns to the  $X_i$  Register.

The 64-bit General Micrand reads the contents of the  $X_i$  Register and supplies it to ALN as the C Operand. The j and k fields enter AC as the B Operand and becomes AC Shift Count. The ALN Functional Unit Micrand enters ALN Control (ALN 1.0) from CST with a value of 8114 1E00 0000 4100<sub>16</sub>.

The C Operand enters the Shift Network Mux (ALN 1.0) and passes on to Shifter Ranks 1,2.

The AC Shift Count, containing the right-shift value, enters the Shift Count Generator (ALN 1.0). The Shift Count proceeds from the Shift Count Generator to Shifter Ranks, 1,2 to control the right shift of the operand.

The Shift Network Output enters the ALN Output Mux (Upper and Lower, ALN 1.0) and returns to the  $X_i$  Register as ALN Result.

1. At micrand sequence Time 31 of Instruction Control Pipeline (ICP) cycle, CST issues Functional Unit Micrand Bits 0 through 63/64 through 71 to ALN (ALN 3.0). Refer to section 3, ICP description, for details of instruction timing in pipeline.
2. Functional Unit Micrand Bit 7 enters ALN Response Control (ALN 3.24) to enable ALN Go to the Holding Register as it arrives at Time 33.

Functional Unit Micrand Bits 8 through 22 enter Micrand Decoders (ALN 3.3). ALN Request enables entry of Micrand Bits 12 through 15 into Mux Control Field Decoder at Time 32.

3. Because Micrand Bits 8 through 11 have a value of 1, Shift Count Generator Control Bits 0, 4, 8, and 9 activate in Shift Count Generator Field Decoder (ALN 3.3).

Because Micrand Bits 16 through 18 have a value of 0, no Mux Complement Control Bits activate.

Because Micrand Bits 19 through 22 have a value of  $F_{16}$ , Enable Shift Parity Check activates.

4. All decoded control bits proceed to Control/Micrand Register input (ALN 3.0).



5. At Time 32, ALN Request activates in ICP and gates Functional Unit Micrand Bits 12 through 15 to Mux Control Field Decoder (ALN 3.3).

Because Micrand Bits 12 through 15 have a value of 4, Mux Control Bits 9, 12, 19, 14, 15, 22, 23, 25, and 27 activate.

6. Mux Control Bits 9, 12, 14, 15, 19, 22, 23, 25, and 27 become Mux Control Bits 7, 10, 11, 21, 13, 17, 25, 22, 23, and 24 (ALN 3.3).
7. Mux Control bits proceed to Control/Micrand Register input (ALN 3.0).
8. At Time 33, ALN Go, a 64-ns signal, arrives at ALN Response Control (ALN 3.24) and data inputs to Control/Micrand Register (ALN 3.0).
9. At Time 40, ALN Response partially enables Clear Micrand Register in Micrand Register Control (ALN 3.24).

ALN Response returns to ICP as an indication ALN is completing operation specified by micrand.

Functional Unit Micrand and decoded control signals enter Control/Micrand Register.

B Operand, containing j and k fields from instruction, enters ALN Shift Count Register (AC 3.14) in bit positions 6 through 11.

Sixty-bit  $X_i$  Register data (sign-extended to 64 bits) enters C Operand Register (OPI 3.11).

Enable Shift Parity Check enters Holding Register (ALN 3.3).

10. AC Shift Count Bits 0 through 11, with j, k fields in bit positions 6 through 11, enter Shift Count Generator (ALN 3.4).

Enable Shift Parity Check permits shift count and micrand parity errors to enter ALN PFS Gates (ALN 3.24).

Shift Count Generator Control Bits 0, 4, 8, and 9 enter Shift Count Generator.

11. Shift Count Generator Control Bits 0 and 9 Select Functional Unit Micrand Bits 0 through 6 and AC Shift Count Bits 6 through 11 for entry into Unbias Networks. Bits 4 and 8 determine that unbiasing is not required. Zeros in Micrand Bits 1 through 6 are added to shift count, which become Shift Count Bits 0 through 5 in Output Control. Micrand Bit 0 becomes Right Shift 64 in Output Control.
12. Mux Control Bits 7, 10, 11, 13, 17, 21 through 25 enter Shift Network Mux (ALN 3.5). In conjunction with Right Shift 64 from Shift Count Generator (ALN 3.4), the bits select C Sign in Shift Network Mux Output bit positions 0 through 63, and C Operand Bits 64 through 127 in bit positions 64 through 127. C Sign is C Operand Sign.
13. Shift Network Mux Bits 0 through 127/128 through 143 enter Shift Network (ALN 3.6).  
  
Functional Unit Micrand Bit 49 selects Shift Count Bits 0 through 5 in Shift Network Muxes.
14. Shift count is decoded to produce right shifts of from 0 through 7 places in Shifter Rank 1 (ALN 3.6) and right shifts of from 0 through 56 places in multiples of eight in Shifter Rank 2 (ALN 3.6).

15. Shift Network Output Bits 64 through 79/136, 137 and Shift Network Output Bits 80 through 127/138 through 143, containing the right-shifted  $X_i$  Register contents depart for ALN Output Mux (ALN 3.10, 3.12).
16. ALN Output Mux selects Shift Network Output bits 64 through 127/136 through 143 for output to OPI as ALN Result Bits 0 through 63/64 through 71. Bits 0 through 3 are sign extension. Bits 4 through 63 comprise data word.
17. Clear Micrand Register activates in Micrand Register Control (ALN 3.24) if ALN Go deactivates in ICP at Time 42. Inactive ALN Go signifies the next micrand is not being issued to ALN.
18. Clear Micrand Register clears Control/Micrand Register (ALN 3.0) before deactivating at Time 50.
19. Right-shifted contents of  $X_i$  Register returns to  $X_i$  Register in Register File (OPI 3.5) at Time 52.

#### C180 INTEGER ADD (24) INSTRUCTION

The instruction adds the contents of the 64-bit  $X_k$  Register in the Register File (OPI 3.5) to the contents of the 64-bit  $X_j$  Register. The instruction supplies the  $j$  and  $k$  designators to the Register File address network. The 64-bit sum returns to the  $X_k$  Register.

The General Micrand controls reading and writing of the X registers. The ALN Functional Unit Micrand controls the summing operation in ALN. The micrand value is 610C 4017 1420 0100<sub>16</sub>.

Data from the  $X_j$  Register becomes the C Operand. Data from the  $X_k$  Register becomes the B Operand. The B Operand passes through the Multiply/Divide Output Mux as B Operand or Multiply/Divide Result and becomes Adder Mux Bits 32 through 95. The C Operand enters the Shift Network Mux and passes through Shifter Ranks 1, 2 to be aligned with Adder Mux Bits 32 through 95. The Shift Network Output and Adder Mux bits enter the Large Adder.

Large Adder Result is selected in the ALN Output Mux (ALN 1.0) for return to the  $X_k$  Register as ALN Result.

1. Refer to C170 Right Shift (21) Instruction, steps 1 and 2, for details of Functional Unit Micrand's arrival in ALN at micrand sequence Time 31.
2. Because Micrand Bits 16 through 18 have a value of 2, Mux Complement Control Bits 2, 4, and 10 activate in Complement Field Decoder (ALN 3.3). These bits become Complement Control Bits 4 through 6, 11, and 12.
3. All decoded control bits proceed to Control/Micrand Register input (ALN 3.0).
4. At Time 32, ALN Request activates in ICP and gates Functional Unit Micrand Bits 12 through 15 to Mux Control Field Decoder (ALN 3.3).

Because Micrand Bits 12 through 15 have a value of  $C_{16}$ , Mux Control Bits 3 and 13, and Enable Integer Path to Adder activate in Mux Control Field Decoder.

5. Mux Control Bits 3 and 13 become Mux Control Bits 26 and 32.
6. Mux Control bits and Enable Integer Path to Adder proceed to Control/Micrand Register input (ALN 3.0).

7. At Time 33, ALN Go, a 64-ns signal, arrives at ALN Response Control (ALN 3.24) and at data inputs to Control/Micrand Register (ALN 3.0).
8. At Time 40, ALN Response partially enables Clear Micrand Register in Micrand Register Control (ALN 3.24).

ALN Response returns to ICP as an indication ALN is completing operation specified by micrand.

Functional Unit Micrand and decoded control signals enter Control/Micrand Register.

Data from 64-bit  $X_k$  Register enters B Operand Register (OPI 3.11).

Data from 64-bit  $X_j$  Register enters C Operand Register (OPI 3.11).

9. B Operand Bits 0 through 63/64 through 71 enter Multiply/Divide Output Mux (ALN 3.16).  
  
C Operand Bits 0 through 63/64 through 71 enter Shift Network Mux (ALN 3.5).
10. B Operand or Multiply/Divide Result Bits 16 through 63/66 through 71 depart for Adder Mux (ALN 3.9). B Operand or Multiply/Divide Result Bits 0 through 15/64, 65 enter Exponent, Integer Mux (ALN 3.16) because Enable Integer Path to Adder in active. Adder Mux Input Bits 0 through 15/16, 17 depart for Adder Mux.
11. C Operand Bits 0 through 63 are duplicated in Shift Network Mux (ALN 3.5) to become C Operand Bits 0 through 63 and 64 through 127.  
  
B Operand Bits 16 through 63 become B Operand Bits 48 through 95 in Adder Mux (ALN 3.9). Adder Mux Input Bits 0 through 15 become B Operand Bits 32 through 47 in Adder Mux.
12. Complement Control Bits 4 through 6, 11, and 12 enter Shift Network Mux and activate Complement B,C Operand Bits 0 through 31, 32 through 39, 40 through 63, 64 through 95, 96 through 127.
13. C Operand Bits 0 through 127 are complemented in Shift Network Mux (ALN 3.5) Complement Control.
14. Inactive Mux Control Bits 0 through 15, 17 through 25 select complemented C Operand Bits 0 through 127 in Shift Network Mux. Values are recomplemented at input to Large Adder (ALN 3.10).  
  
Mux Control Bits 26 and 32 select B Operand sign and B Operand Bits 32 through 95 in Adder Mux (ALN 3.9). B Operand Sign [(B Operand Bit 0 from B Operand Sign Selector (ALN 3.14))] is fanned out to Adder Mux Bits 0 through 31.
15. Shift Network Mux Bits 0 through 127/128 through 143 enter Shift Network (ALN 3.6).
16. Inactive Micrand Bits 49, 50 select Micrand Bits 1 through 6 in shift count muxes (ALN 3.6).
17. Constant of  $48_{10}$  in Micrand Bits 1 through 6 right-shifts Shift Network Mux Bits 0 through 127/128 through 143, 48 places end-off and sign-extended.

18. Shift Network Output Bits 16 through 111/130 through 141 (complemented C Operand sign in bit positions 16 through 47, and complemented C Operand Bits 0 through 63 in bit positions 48 through 111) enter Large Adder (ALN 3.10), where they are recomplemented and become ALU input bit 0 through 95.

Adder Mux Bits 0 through 95/96 through 107 enter Large Adder ALU.

19. Inactive Enable Normalize Count to Adder and Micrand Bits fields 26, 27 and 28, 29 (each having a value of 1) enable a 97-bit summing operation in the ALU. Micrand Bits 30 through 32 (with a field value of 6) limit the add to 64 bits and specify two's complement operation.
20. Large Adder Result Bits 48 through 95/102 through 107 enter ALN Output Mux, Bits 16 through 63/Parity (ALN 3.10) because Micrand Bit 36, 37 field equals 1. ALN Result Bits 16 through 63/66 through 71 proceed to  $X_k$  Register in Register File (OPI 3.5).  
  
Large Adder Result Bits 32 through 47/100, 101 enter ALN Output Mux, Bits 0 through 15, Parity (ALN 3.12) because Micrand Bit 34, 35 field equals 1. ALN Result Bits 0 through 15/64, 65 proceed to  $X_k$  Register.
21. Clear Micrand Register activates in Micrand Register Control (ALN 3.24) if ALN Go deactivates is ICP at Time 43. Inactive ALN Go signifies next micrand is not being issued to ALN.
22. Clear Micrand Register clears Control/Micrand Register (ALN 3.0) before deactivating at Time 50.
23. ALN result enters  $X_k$  Register at Time 52.

#### C180 SINGLE-PRECISION FLOATING-POINT SUM (30) INSTRUCTION

This instruction compares the exponents of operands from the  $X_j$  Register and the  $X_k$  Register in the Register File (OPI 3.5). If the exponents are not equal, the instruction aligns the coefficients by right-shifting the one with the smaller exponent the number of places equal to the difference between the two exponents. The maximum shift which does not produce loss of significance is 48 bit positions.

The sum of the two 48-bit coefficients returns to the  $X_k$  Register, along with the larger exponent. The instruction issues two ALN Functional Unit Micrands consecutively. The micrands values are 01AA8455A80A6102<sub>16</sub> and 1FF9C697258F2100<sub>16</sub>. The first produces the result; the second normalizes it.

During the first major cycle (first micrand) the contents of the  $X_k$  Register become the B Operand. The contents of the  $X_j$  Register become the C Operand. B and C Operand Bits 0 through 15 enter the Shift Count Generator (ALN 1.0) at the same time B and C Operand Bits 1 through 15 enter the Exponent Arithmetic Network. The B Operand enters the Multiply/Divide Output Mux and becomes B Operand or Multiply/Divide Result.

The C Operand enters the Shift Network Mux (ALN 1.0) and the Adder Mux. The B Operand or Multiply/Divide Result (containing the B Operand) also enters the Shift Network Mux and the Adder Mux.

The Shift Count Generator compares the two operand exponents and selects the coefficient with the larger exponent in the Adder Mux. Active or inactive C Operand Greater than B Operand performs the selection. The same signal selects the coefficient with the smaller exponent in the Shift Network Mux.

The difference between the two exponents becomes the Shift Count produced in the Shift Count Generator. Shift Network Mux bits are right-shifted in Shifter Ranks 1, 2 (ALN 1.0) the amount of the shift count.

Shift Network Output and Adder Mux coefficients enter the Large Adder (ALN 1.0), which produces the coefficient sum. Large Adder Result is selected in ALN Output Mux (Lower) to become ALN Result (Lower).

Active or inactive C Operand Greater than B Operand selects the larger exponent, in the Exponent Arithmetic Network (ALN 1.0). Exponent Network Result proceeds through Normal/Endcase Exponent Selector to become ALN Result (Upper) in ALN Output Mux (Upper). This intermediate result enters a scratch register in the Register File (OPI).

During the second major cycle (second micrand) the shortstopped ALN Result from previous cycle becomes the C Operand.

The C Operand enters the Shift Network Mux (ALN 1.0) and the Normalize Encoder.

The Significance Count formed in the Normalize Encoder enters Shifter Ranks 1, 2 to left-shift an unnormalized coefficient to the normalized position. The count also may right-shift the coefficient one place if overflow was detected in the previous cycle. The Normalize Count adjusts the incoming exponent in the Exponent Arithmetic Network to account for modification of the coefficient.

Shift Network Output (normalized coefficient) passes through the Large Adder (ALN 1.0) and enters ALN Output Mux (Lower). The adjusted exponent enters ALN Output Mux (Upper). ALN Result enters the  $X_k$  Register.

1. Refer to C170 Right Shift (21) Instruction, steps 1 and 2, for details of first Functional Unit Micrand's arrival in ALN at micrand sequence Time 31.
2. Because Micrand Bits 8 through 11 have value of A, Shift Count Generator Control Bits 4, 5, 6, 8, 9, 11, and 13 activate in Shift Count Generator Field Decoder (ALN 3.3).

Because Micrand Bits 16 through 18 have a value of 4, Mux Complement Control Bits 0, 1, 5 through 8 activate in Complement Field Decoder (ALN 3.3). These bits become Complement Bits 7 and 8 and Complement Control Bits 0 through 3, 7 through 10.

Because Micrand Bits 19 through 22 have a value of 2, Exponent Network Control Bits 0, 6, 8, 9, and 11 activate in Exponent Arithmetic Field Decoder (ALN 3.3).

3. All decoded control bits proceed to Control/Micrand Register input (ALN 3.0).
4. At Time 32, ALN Request activates in ICP and gates Functional Unit Micrand Bits 12 through 15 to Mux Control Field Decoder (ALN 3.3).

Because Micrand Bits 12 through 15 have a value of A<sub>16</sub>, Mux Control Bits 0 through 3, 5, 6, 11 through 14, 19 through 21, 23 through 29, 31, 32 activate.

5. Mux Control Bits 0 through 3, 5, 6, 11 through 13, 19 through 21, 23 through 29, 31 and 32 become Mux Control Bits 0 through 4, 6, 8 through 12, 17 through 24, 26 through 30, 32, 33, 36 through 38, 40, 41.
6. Mux Control bits proceed to Control/Micrand Register input (ALN 3.0).
7. At Time 33, ALN Go, a 64-ns signal, arrives at ALN Response Control (ALN 3.24) and at data inputs to Control/Micrand Register (ALN 3.0).

8. At Time 40, ALN Response partially enables Clear Micrand Register in Micrand Register Control (ALN 3.24).

ALN Response returns to ICP as indication ALN is completing operation specified by micrand.

Functional Unit Micrand and decoded control signals enter Control/Micrand Register.

Data from 64-bit  $X_k$  Register in Register File (OPI 3.5) enters B Operand Register (OPI 3.11).

Data from 64-bit  $X_j$  Register enters C Operand Register (OPI 3.11).

9. B Operand Bits 0 through 15 and C Operand Bits 0 through 15 enter Shift Count Generator (ALN 3.4).

B Operand Bits 1 through 15 and C Operand Bits 1 through 15 enter Exponent Arithmetic Network (ALN 3.8).

B Operand Bits 0 through 63/64 through 71 enter Multiply/Divide Output Mux (ALN 3.16).

C Operand Bits 0 through 63/64 through 71 enter Shift Network Mux (ALN 3.5).

C Operand Bits 16 through 63/66 through 71 enter Adder Mux (ALN 3.9).

10. Shift Count Generator Control Bits 5, 6, and 9 select B Operand Bits 1 through 15 in Mux A and C Operand Bits 1 through 15 in Mux B of Shift Count Generator (ALN 3.4).

Shift Count Generator Control Bits 4 and 8 select biased exponent values in Unbias Network A and B.

Shift Count Generator Control Bit 11 complements output of Mux B entering Adder.

Adder subtracts<sup>4</sup> C Operand Bits 1 through 15 from B Operand Bits 1 through 15.

Shift Count Generator Control Bit 12 enables selection of Adder Out Bits 1 through 8 and absolute value of bits 9 through 15 in Output Control. Bits 1 through 8 represent sign of difference. If negative, C Operand Greater than B Operand activates. Absolute value of bits 9 through 15 represents the exponent difference in magnitude form.

Shift Count Generator Control Bit 13 and Micrand Bit 44 activate Shift Fault Lower in Error Detector, which becomes Force Bits 64 through 127 to Zero or Sign.

11. Exponent Network Control Bit 0 enters Exponent Arithmetic Network (ALN 3.8) to select B Operand Bits 1 through 15 and C Operand Bits 1 through 15 as Mux A and Mux B Bits 1 through 15, respectively.

Exponent Network Control Bits 8, 9 permit Mux A and B Bits 1 through 15 to pass through Unbias Networks A and B in biased form.

Exponent Network Control Bit 6 forces Mux A or B Bits 1 through 15 to zero, depending state of C Larger Than B. Effect is to pass larger exponent through Adder unchanged.

Output Control selects without modification Adder Output Bits 1 through 15 to become Exponent Network Result Bits 1 through 15.

12. B Operand or Multiply/Divide Result Bits 48 through 63/70, 71 enter Exponent Integer Mux (ALN 3.16) to become Adder Mux Input Bits 0 through 15/16, 17.
13. B Operand or Multiply/Divide Result Bits 16 through 47/66 through 69 enter Adder Mux (ALN 3.9) along with Adder Mux Input Bits 0 through 15/16, 17 from Exponent, Integer Mux.
14. Complement Bits 7 and 8 place magnitude form coefficients in signed form in Complementer.

Mux Control Bits 26 through 30, 32, 33, 36 through 38, 40, and 41 select coefficient with larger exponent in Adder Mux, based on state of C Larger Than B.

Adder Mux Bits 0 through 95/96 through 107 contain signed coefficient in bit positions 0 through 47 and zeros in bit positions 48 through 95.

15. B Operand or Multiply/Divide Result Bits 0 through 63/64 through 71 enter Shift Network Mux (ALN 3.5).
16. Complement Control Bits 0 through 3, 7 through 10 complement B Operand and C Operand in Complement Control (ALN 3.5) if their signs are positive. Shift Network Output (ALN 3.6) is complemented at input to Large Adder (ALN 3.10). Since two complements occur along this path, the first complement effectively produces signed equivalent of magnitude form B or C Operand coefficients.  
  
Mux Control Bits 0 through 4, 6, 8 through 12, 17 through 24 (Mux Code equals A) and C Operand Greater Than B Operand select B or C Operand coefficient with smaller exponent. If magnitude of shift count from Shift Count Generator (ALN 3.4) exceeds 63, activated Right Shift 64 right-shifts selected coefficient 64 places before it enters Shift Network (ALN 3.6).
17. Shift Network Mux Bits 0 through 127/128 through 143 enter Shift Network (ALN 3.6).
18. Shift Count Bits (value  $\leq 63$ ) enter shift count Muxes because Micrand Bits 49, 50 have a coded value of 3.

Shift Network Mux Bits 0 through 127/128 through 143 are right-shifted designated amount in Shifter Ranks 1 and 2 (ALN 3.6). Effect of shift is to align coefficient having smaller exponent with coefficient having larger exponent.

Force Bits 64 through 127 To Zero or Sign and Micrand Bits 49, 50 gate Extended Sign to Shift Network Output Bits 64 through 127/136 through 143. This instruction is a single-precision operation. Only Shift Network Output Bits 16 through 63 are used in Large Adder arithmetic operation which follows.

19. Large Adder ALU (ALN 3.10) performs 97-bit one's complement addition of Adder Mux Bits 0 through 95 and complemented Shift Network Out Bits 16 through 111. Micrand Bits 30 through 32 have decoded value of 3. Micrand Bit 26, 27 and 28, 29 fields equal 1.
20. Biased Exponent Network Result Bits 1 through 7 enter C170/C180 Exponent Format Select (ALN 3.11) because Exponent Network Control Bit 2 is inactive. Internal Sign, an invalid sign, occupies coefficient sign position in C180 format exponent. Normal Exponent Bits 0 through 7/64 proceed to ALN Output Mux, Bits 0 through 15/Parity (ALN 3.12).

Exponent Network Result Bits 8 through 15 enter Normal/Endcase Exponent Selector (ALN 3.11) to become Exponent Bits 8 through 15/65; provided C180 exponent overflow condition does not exist. Exponent Bits 8 through 15/65 proceed to ALN Output Mux, Bits 0 through 15/Parity.

21. ALN Output Mux, Bits 16 through 63/Parity (ALN 3.10) selects single-precision coefficient addition result (Large Adder Result Bits 0 through 47/96 through 101). Result is in signed form.

ALN Output Mux, Bits 0 through 15/Parity selects Normal Exponent Bits 0 through 7/64 and Exponent Bits 8 through 15/65. Micrand Bit 34 is a logical one.

22. ALN Result Bit 0 through 63/64 through 71 depart for Register File location 08 (OPI 3.5).
23. ALN result enters Register File location 08 at Time 52. ALN Result is gated through operand short stop path in OPI (OPI 3.11) for entry into C Operand Register (OPI 3.11) at second micrand's Time 40. B Operand Register (OPI 3.11) is zero-filled at Time 40.
24. Refer to C170 Right Shift (21) instruction, steps 1 and 2, for details of second Functional Unit Micrand's arrival in ALN at Time 31 (Time 41 of first micrand).
25. Because Micrand Bits 8 through 11 have a value of F<sub>16</sub>, Shift Count Generator Control Bits 4, 8, 11, 16, and 18 activate in Shift Count Generator Field Decoder (ALN 3.3).

Because Micrand Bits 16 through 18 have a value of 6, Mux Complement Control Bits 0, 1, 5, and 6 activate in Complement Field Decoder (ALN 3.3). These bits become Complement Control Bits 0 through 3, 7 through 10.

Because Micrand Bits 19 through 22 have a value of 3, Exponent Network Control Bits 1, 4, 8, 9, and 12 activate in Exponent Arithmetic Field Decoder (ALN 3.3).

26. All decoded control bits proceed to Control/Micrand Register input (ALN 3.0).
27. At Time 32 (Time 42 of first micrand), ALN Request activates in ICP and gates Functional Unit Micrand Bits 12 through 15 to Mux Control Field Decoder (ALN 3.3).  
Because Micrand Bits 12 through 15 have a value of 9, Mux Control Bits 0, 2, 12, 14, 19, 22, and Enable Integer Path to Adder activate in Mux Control Field Decoder (ALN 3.3).
28. Mux Control Bits 0, 2, 12, 14, 19, and 22 become Mux Control Bits 0, 3, 4, 6, 10, 11, 17, 21, and 25.
29. Mux Control bits and Enable Integer Path to Adder proceed to Control Micrand Register input (ALN 3.0).
30. At Time 33, ALN Go, a 64-ns signal, arrives at ALN Response Control (ALN 3.24) and at data inputs to Control Micrand Register (ALN 3.0).
31. At Time 40 (first micrand's Time 50), ALN Response partially enables Clear Micrand Register in Micrand Register Control (ALN 3.24).

ALN Response returns to ICP as indication ALN is completing operation specified by micrand.

Functional Unit Micrand and decoded control signals enter Control/Micrand Register. Literal zeros enters B Operand Register (OPI 3.11).

Shortstopped ALN Result from first micrand enters C Operand Register (OPI 3.11).



If overflow occurred in Large Adder in step 19, Large Adder Overflow enters Control/Micrand Register.

Internal Sign, formed from Adder Result Sign (step 19) and Micrand Bit 25 of first micrand, enters Control/Micrand Register.

32. C Operand Bits 0 through 63/64 through 71 enter Shift Network Mux (ALN 3.5). Bits 0 through 63 also enter Normalize Encoder (ALN 3.7). Because coefficient may be in complemented form, Shift Count Generator Control Bit 16 enables Internal Sign, the preserved Large Adder Result Sign, to complement returning coefficient prior to leading zero count.  
  
C Operand Bits 1 through 15 enter Exponent Arithmetic Network (ALN 3.8).  
  
B Operand Bits 0 through 63/64 through 71 (zeros) enter Multiply/Divide Output Mux (ALN 3.16).
33. B Operand or Multiply/Divide Result Bits 0 through 15/16, 17 (zeros) enter Exponent, Integer Mux (ALN 3.16) because Enable Integer Path to Adder is active.
34. Adder Mux Input Bits 0 through 15/16, 17 and B Operand or Multiply/Divide Result Bit 16 through 63/66 through 71 proceed to Adder Mux (ALN 3.9).
35. Micrand Bits 23 through 25 equal to 2 activates one of three signals in Normalize Encoder translator. If Overflow from Control/Micrand Register (ALN 3.0) is inactive and significant data is present in ALN add result contained in C Operand, Force Bits 1 through 15 to Zero, Count Leading Zeros, Bits 1 through 63 activates. If no significant data is present and Overflow is inactive Force Count to  $OF_{16}$  activates. If Overflow is active, Force Count To  $OE_{16}$  activates.
36. Significance Count Bits 0 through 5 proceed to Shift Network (ALN 3.6).  
  
Significance Encoder Bits 0 through 5 enter Significance Count Adjustment Adder (ALN 3.7) where they are added to Add One to Exponent and Micrand Bits 0 through 6 (value of  $OF_{16}$ ). If C Operand Bits 1 through 15 are forced to zeros for leading zero count in Normalize Encoder, constant  $OF_{16}$  adjusts leading zero count to produce a true normalizing value for exponent arithmetic. Principle also applies to forced leading zero counts when no significant data is present and to overflow conditions. In these cases, constant adjusts forced leading zero count to a true normalizing value for exponent arithmetic.
37. Normalize Count Bits 1 through 15 proceed to Exponent Arithmetic Network (ALN 3.8).
38. Complement Control Bits 0 through 3, 7 through 10 enable complementing of C Operand Bits 0 through 127 in Shift Network Mux (ALN 3.5) if C Operand Sign is positive. C Operand Sign is preserved Internal Sign from Large Adder (ALN 3.10) result of previous operation. Shift Count Generator Control Bit 16 selects Internal Sign in C Operand Sign Selector (ALN 3.13). Because output of Shift Network is complemented entering Large Adder, effect of complementing positive operand in Shift Network Mux is to leave the operand uncomplemented for Large Adder operations. Opposite applies for negative operands. Effect is to produce magnitude form coefficient in Large Adder.  
  
Mux Control Bits 0, 3, 4, 6, 10, 11, 17, 21 and 25 select zeros in bit positions 0 through 7, Overflow or Floating-Point Coefficient Sign in bit positions 8 through 15, C Operand Sign in bit positions 64 through 127 and C Operand Bits 16 through 63 (coefficient in magnitude form) in bit positions 16 through 63. Shift Network Mux Bits 0 through 127/128 through 143 proceed to Shift Network (ALN 3.6).

NOTE

Shift Count Generator Control Bit 18 enables Overflow or Adder Result Sign in Shift Mux Sign Selector (ALN 3.4). C Operand Sign (ALN Result Sign) places Overflow or Adder Result Sign in proper state to force a logical one in bit 15 position of Shift Network Mux (ALN 3.5).

39. Adder Mux (ALN 3.9) selects B Operand Bits 0 through 95. B Operand Bits 0 through 95 include Adder Mux Input Bits 0 through 15/16, 17 and B Operand or Multiply/Divide Result Bits 16 through 63/66 through 71 (bits 16 through 47 are fanned out to bit positions 0 through 31). Adder Mux Bits 0 through 95/96 through 107 (equal to zero) proceed to Large Adder (ALN 3.10).

40. Because Exponent Network Control Bits 1, 4, 8, 9, and 12 are active, Normalize Count Bits 1 through 15 and C Operand Bits 1 through 15 become Mux A Bits 1 through 15 and Mux B Bits 1 through 15, respectively, in Exponent Arithmetic Network (ALN 3.8). C Operand Bits 1 through 15 are biased exponent produced in previous operation. Normalize Count Bits 1 through 15 are exponent adjustment for left-shifting coefficient to normalize it, for right-shifting coefficient one place to accommodate overflow bit, or for no shift condition that results from absence of significant data in coefficient.

Mux A, B Bits 1 through 15 pass through unbiased and complement networks unchanged. Two's complement addition occurs in Adder to produce normalized exponent.

Exponent Network Result Bits 1 through 15 proceed to exponent selectors (ALN 3.11).

41. Active Micrand Bit 50 selects complement of Significance Count Bits 0 through 5 in Shift Network Mux's (ALN 3.6).

Complement of Significance Count right-shifts C Operand coefficient in Shift Network Mux Bits 16 through 63 amount required to place most significant Shift Network Mux Bit in bit 64 position of Shift Network output. If no significant data is present, Shift Network Mux Bit 16 enters bit 64 position. In event overflow condition exists, right shift places Shift Network Mux Bit 15 (logical one) in bit 64 position.

Shift Network Output Bits 16 through 111/130 through 141, proceed to Large Adder (ALN 3.10). Coefficient is in one's complement form in bit positions 64 through 111.

42. Adder Mux Bits 32 through 95 (equal to zero) and recomplemented Shift Network Output Bits 48 through 111 enter Large Adder ALU where 64-bit two's complement addition is performed.

Normalized floating-point coefficient in magnitude form occupies Large Adder Result bit positions 48 through 95.

Large Adder Result Bits 48 through 95/102 through 107 proceed to ALN Output Mux, Bits 16 through 63/Parity (ALN 3.10).

43. Exponent Network Result Bits 1 through 7 enter C170/180 Exponent Format Selector (ALN 3.11), because Exponent Network Control Bit 2 is inactive. Exponent Network Result Bits 8 through 15 enter Normal/Endcase Exponent Selector (ALN 3.11).

If significant data has been detected in Normalize Encoder (ALN 3.7), Internal Sign becomes bit 0 of C180 format Normal Exponent. (Internal Sign is Large Adder coefficient result sign from first micrand operation, which produced coefficient sum.) If no significant data is present, Normal Exponent Bit 0 is a logical 0. Bit 0 is coefficient sign for C180 floating-point operand.

Normal Exponent Bits 0 through 7/64 and Exponent Bits 8 through 15/65 proceed to ALN Output Mux, Bits 0 through 15/Parity (ALN 3.12).

44. Because Micrand Bit 37 is active, Large Adder Result Bits 48 through 95/102 through 107 become ALN Result Bits 16 through 63/66 through 71 in ALN Output Mux, Bits 16 through 63/Parity (ALN 3.10).

Because Micrand Bit 34 is active, Normal Exponent Bits 0 through 7/64 and Exponent Bits 8 through 15/64, 65 become ALN Result 0 through 15/64, 65 in ALN Output Mux, Bits 0 through 15/Parity (ALN 3.12).

45. Refer to C180 Integer Add (24) instruction description, steps 21 through 23, for final events of instruction execution.

If no significant data is detected by the Normalize Encoder (ALN 3.7) during second major cycle, Single-Precision Loss of Significance activates in the Loss of Significance Detector (ALN 3.21). Because the error code contained in Micrand Bits 40 through 42 equals 4, Enable Floating-Point Loss of Significance is active (refer to Exception Signal Select Decoder ALN 3.23). Floating-Point Loss of Significance proceeds to ICC, where it becomes User Condition Register Bit 12.

Single-Precision Loss of Significance and inactive User Mask Register Bit 12 (a condition established at the discretion of the user) activate Clear Exponent Bits in Exponent Zero Control (ALN 3.21).

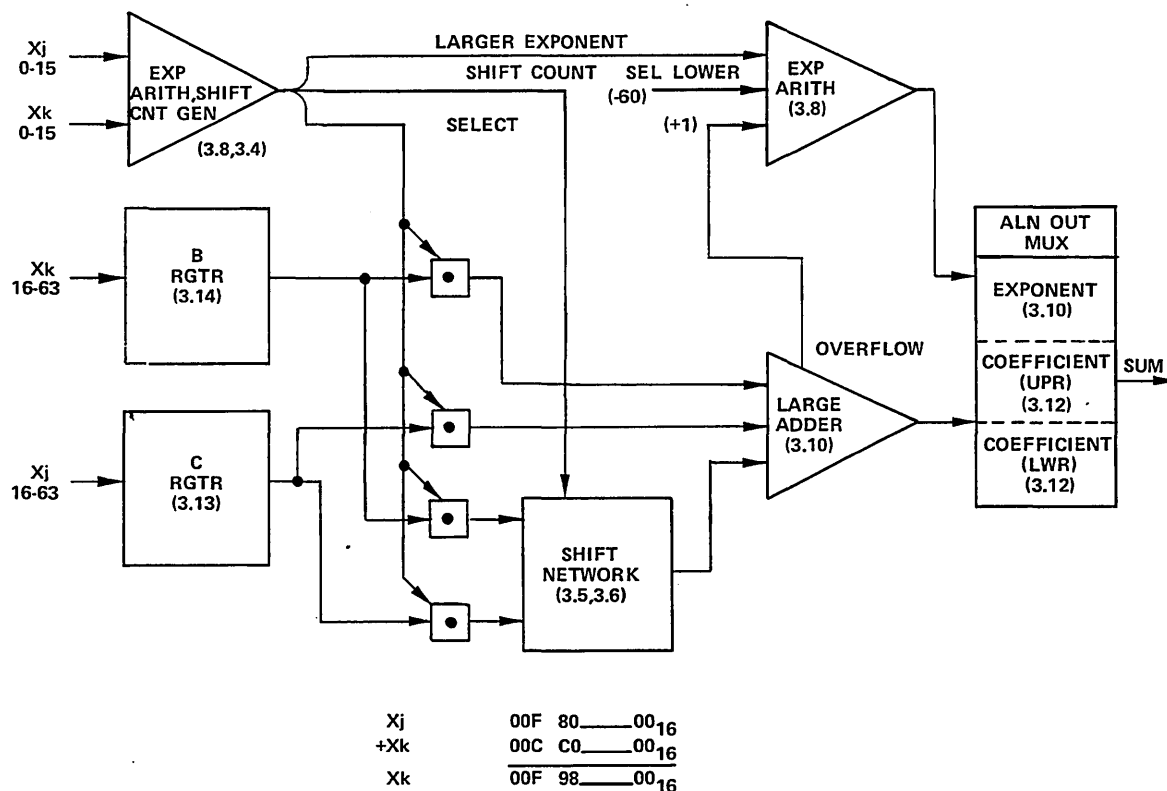
Clear Exponent Bits activates Force ALN Output Bits 16 through 31, 40 through 63 to Zero and Force ALN Output Bits 32 through 39 to Zero in Endcase Coefficient Control (ALN 3.22). Clear Exponent Bits also activates Disable ALN Output Bit 0 through 15 in Disable ALN Output Translator (ALN 3.12). Disable ALN Output Bits 16 through 31, 40 through 63 and Disable ALN Output Bits 32 through 39 also activate in same translator. The entire ALN result is forced to zero.

If the User Mask Register bit is set by software, the floating-point result returns to OPI unmodified.

Forced floating-point operand values may also be produced in the event an exponent endcase condition exists, that is, infinite, indefinite exponent values. This subject is analyzed in Endcase Conditions.

Figure 4-11 illustrates C180 single-precision floating-point addition.

Figure 4-12 summarizes the arithmetic steps required to perform the double-precision equivalent of the 30 instruction.



1. DETERMINE LARGER EXPONENT. ( $X_j \text{ EXP} - X_k \text{ EXP}$ )

$$\begin{array}{r} 00F \\ -00C \\ \hline 003 = X_j \text{ LARGER, SHIFT COUNT} = 3. \end{array}$$

2. EQUALIZE EXPONENTS. (RIGHT SHIFT SMALLER EXPONENT'S COEFFICIENT.)

$$X_k = C0 \text{---} 00_{16} \text{ (RIGHT SHIFT 3)} = 18 \text{---} 00_{16}$$

3. ADD COEFFICIENTS.
 
$$\begin{array}{r} 80 \text{---} 00_{16} \\ 18 \text{---} 00_{16} \\ \hline 98 \text{---} 00_{16} \end{array}$$

4. RESULT EQUALS MODIFIED LARGER EXPONENT AND COEFFICIENT SUM.

$$X_k = 00F \ 98 \text{---} 00_{16}$$

5. THE SUM OR DIFFERENCE RE-ENTERS THE C REGISTER AND THE SHIFT AND EXPONENT ARITHMETIC NETWORKS DURING A SECOND PASS TO PROVIDE A NORMALIZED RESULT.

Figure 4-11. C180 Single-Precision Floating-Point Addition/Subtraction

$$(X_j, X_j + 1) + (X_k, X_k + 1) = X_k, X_k + 1$$

$$\begin{array}{ll} X_j = 27 \times 2^0 & X_j + 1 = 22 \times 2^{-48} \\ X_k = 35 \times 2^0 & X_k + 1 = 22 \times 2^{-48} \\ X_k = 62 \times 2^0 & X_k + 1 = 44 \times 2^{-48} \end{array}$$

① MOVE  $X_j + 1$  (WITH  $X_j$  EXP -48) TO SCRATCH REGISTER 2

② MOVE  $X_k + 1$  (WITH  $X_k$  EXP -48) TO SCRATCH REGISTER 1

③  $X_j + X_k =$  SCRATCH REGISTER 3 (UPPER)

$$\begin{array}{r} 2700 (0) \\ 3500 (0) \\ \hline 6200 (0) \end{array}$$

④ SCRATCH REGISTER 2 + SCRATCH REGISTER 1 =  
SCRATCH REGISTER 4 (UPPER)

$$\begin{array}{r} 2200 (-48) \\ 2200 (-48) \\ \hline 4400 (-48) \end{array}$$

⑤ SCRATCH REGISTER 2 + SCRATCH REGISTER 1 =  
SCRATCH REGISTER 2 (LOWER)

$$\begin{array}{r} 2200 (-48) \\ 2200 (-48) \\ \hline 4400 (-96) \end{array}$$

⑥  $X_j + X_k =$  SCRATCH REGISTER 1 (LOWER)

$$\begin{array}{r} 2700 (0) \\ 3500 (0) \\ \hline 6200 (-48) \end{array}$$

⑦ SCRATCH REGISTER 4 + SCRATCH REGISTER 1 =  
SCRATCH REGISTER 5 (LOWER)

$$\begin{array}{r} 4400 (-48) \\ 0000 (-48) \\ \hline 4400 (-96) \end{array}$$

⑧ SCRATCH REGISTER 4 + SCRATCH REGISTER 1 =  
SCRATCH REGISTER 4 (UPPER)

$$\begin{array}{r} 4400 (-48) \\ 0000 (-48) \\ \hline 4400 (-48) \end{array}$$

⑨ SCRATCH REGISTER 4 + SCRATCH REGISTER 3 =  
SCRATCH REGISTER 1 (UPPER)

$$\begin{array}{r} 4400 (-48) = 0044 (0) \\ 6200 (0) \quad 6200 (0) \\ \hline 6244 (0) \end{array}$$

⑩ SCRATCH REGISTER 4 + SCRATCH REGISTER 3 =  
SCRATCH REGISTER 4 (LOWER)

$$\begin{array}{r} 4400 (-48) = 0044 (0) \\ 6200 (-0) \quad 6200 (0) \\ \hline 6244 (-48) \end{array}$$

⑪ SCRATCH REGISTER 4 + SCRATCH REGISTER 5 =  
SCRATCH REGISTER 4 (UPPER)

$$\begin{array}{r} 4400 (-48) = 4400 (-48) \\ 0000 (-96) \quad 0000 (-48) \\ \hline 4400 (-48) \end{array}$$

⑫ SCRATCH REGISTER 2 + SCRATCH REGISTER 4 =  
SCRATCH REGISTER 4 (UPPER)

$$\begin{array}{r} 0000 (-96) = 0000 (-48) \\ 4400 (-48) \quad 4400 (-48) \\ \hline 4400 (-48) \end{array}$$

⑬ SCRATCH REGISTER 1 + SCRATCH REGISTER 4 =  
SCRATCH REGISTER 2 (LOWER)

$$\begin{array}{r} 6200 (0) = 6200 (0) \\ 4400 (-48) \quad 0044 (0) \\ \hline 6244 (-48) \end{array}$$

⑭ SCRATCH REGISTER 1 + SCRATCH REGISTER 4 =  
SCRATCH REGISTER 1 (UPPER)

$$\begin{array}{r} 6200 (0) = 6200 (0) \\ 4400 (-48) \quad 0044 (0) \\ \hline 6244 (0) \end{array}$$

⑮ SCRATCH REGISTER 2 + SCRATCH REGISTER 1 =  
 $X_k$  (UPPER)

$$\begin{array}{r} 4400 (-48) = 0044 (0) \\ 6200 (0) \quad 6200 (0) \\ \hline 6244 (0) \end{array}$$

⑯ SCRATCH REGISTER 2 + SCRATCH REGISTER 1 =  
 $X_k + 1$  (LOWER)

$$\begin{array}{r} 4400 (-48) = 0044 (0) \\ 6200 (0) \quad 6200 (0) \\ \hline 6244 (48) \end{array}$$

Figure 4-12. C180 Double-Precision Floating-Point Sum (34) Instruction

#### C170 SINGLE-PRECISION FLOATING-POINT PRODUCT (40) INSTRUCTION

This instruction forms the floating-point product of operands from the  $X_j$  and  $X_k$  Registers in the Register File (OPI).

The operands are unpacked and the exponents added to form the exponent of the result. The coefficients are multiplied to create a 96-bit product. The upper half of the 96-bit product (single-precision) becomes the coefficient of the result. The result enters the  $X_i$  Register after the coefficient and exponent are packed.

If the multiplier and the multiplicand are normalized and the product only has 95 significant bits, the instruction normalizes it. If either input is unnormalized, no attempt is made to normalize the product.

The instruction issues three ALN Functional Unit Micrands, with values of 01BE 6000 0010 0200<sub>16</sub>, 01FE 0F00 2218 0014<sub>16</sub>, and 7DF7 C2D7 251F 20E4<sub>16</sub>.

During the first major cycle (first micrand), the contents of the  $X_j$  Register become the B Operand. The contents of the  $X_k$  Register become the C Operand.

The C Operand coefficient (bits 16 through 63) enters the C Register (ALN 1.0) as the multiplier. The B Operand coefficient enters the B Register as the multiplicand. The B and C Operand coefficient signs are extended through 16 high-order bit positions in each register.

Multiply/Divide Control operates on minor cycle boundaries to produce 12 bits of final product each 16 ns. The Multiplicand and the low-order 13 bits of Multiplier Data enter Product Networks A, B each minor cycle (ALN 1.0). Multiply Partial Sum and Carry values proceed from the Multiply Partial Summing Network to the Multiply Final Adder. Twelve bits of the Multiply Result enter the C Register each minor cycle in the high-order bit positions. The balance of the multiplier is right-shifted 12 bit positions in the C Register each minor cycle.

During the second major cycle, the multiply operation continues with the contents of the  $X_j$  and  $X_k$  Registers reissued to the B and C Register, respectively.

B Operand Bits 1 through 15 and C Operand Bits 1 through 15 enter the Exponent Arithmetic Network (ALN 1.0). The two are added to form the result exponent, which becomes ALN Result (Upper) in the ALN Output Mux (ALN 1.0).

By the beginning of the third major cycle, the multiply operation is complete. The single-precision product coefficient (high-order 48 bit of 96-bit product) proceeds from the Multiply Final Adder (ALN 1.0) to the Multiply/Divide Output Mux as the Multiply Result. B Operand or Multiply/Divide Result enters the Shift Network Mux.

If normalization is required, the appropriate Significance Count enters shifter Ranks 1, 2 (ALN 1.0) and the appropriate Normalize Count enters the Exponent Arithmetic Network.

Shift Network Mux bits, containing the single-precision coefficient, enter Shifter Ranks, 1, 2, where a one-bit left-shift occurs if normalization is needed. The Shift Network Output proceeds through the Large Adder to the ALN Output Mux (Lower).

The shortstopped product exponent from the previous major cycle becomes C Operand Bits 1 through 15, which return to the Exponent Arithmetic Network (ALN 1.0).

The Exponent Arithmetic Network forms the final Exponent Network Result, with adjustments for normalizing and for the fact a single-precision coefficient has an exponent which is larger by 48 than the 96-bit product exponent. The Exponent Network Result enters ALN Output Mux (Upper).

ALN Result proceeds to the  $X_j$  Register in the Register File.

1. Refer to C170 Right Shift (21) Instruction, steps 1 and 2, for details of first Functional Unit Micrand's arrival in ALN at micrand sequence Time 31.

Additionally, Micrand Bits 43, 49, 50, 53 through 55, and 61 proceed directly to Multiply/Divide Minor and Major Cycle Control Registers (ALN 3.2). Micrand Bits 8 through 11, 16 through 18, 60, and 62 are fanned out from Micrand Decoders (ALN 3.3) to Multiply/Divide Major Cycle Control Register. Micrand Bits 56 through 59 enter Soft Control Memory Address Selector (ALN 3.1).

2. At Time 32, ALN Request activates in ICP and gates Micrand Bits 12 through 15 to Mux Control Field Decoder (ALN 3.3).

Because Micrand Bits 12 through 15 have a value of  $E_{16}$ , Mux Control Bits 10, 24, 26, and 30 activate, becoming Mux Control Bits 31, 36, 37, and 39.

Soft Control Address Bits 0 through 5 (with a value of zero) enter Soft Control Memory Address Register (ALN 3.1). Soft Control Address Bits 4, 5 from Soft Control Address Bit Generator (ALN 3.1) are equal to zero during PH2.

Inactive Micrand Bit 55 enters Holding Register in Multiply/Divide Minor Cycle Control Register (ALN 3.2).

3. Mux Control bits proceed to Control/Micrand Register input (ALN 3.0).

Soft Control Memory Address Bits 0 through 5 address location zero in Soft Control Memory RAM (ALN 3.1). Read Data Bits 0 through 31/32 through 35 proceed to Soft Control Data Out Register (ALN 3.1).

4. At Time 33, ALN Go, a 64-ns signal, arrives at ALN Response Control (ALN 3.24) and data inputs to Control/Micrand Register (ALN 3.0).

Read Data Bits 0 through 31/32 through 35 enter Soft Control Data Out Register (ALN 3.1) to become Soft Control Data Out Bits 0 through 31/32 through 35.

Soft Control Memory Address Bits 0 through 5 address location one of Soft Control Memory RAM (ALN 3.1). Second Read Data word proceeds to Soft Control Data Out Register.

Inactive Micrand Bit 55 enters 16-ns Delay in Multiply/Divide Minor Cycle Control Register (ALN 3.2).

ALN Request enters Holding Register (ALN 3.3).

5. ALN Request and inactive Micrand Bit 55 select Soft Control Data Out Bits 0 through 15/32, 33 in Multiply/Divide Minor Cycle Control Register.

The 16 selected bits provide soft control for first minor cycle of multiply operation. Soft Control bit assignments appear in ALN Control, ALN Control Field Descriptions.

#### NOTE

Micrand Bits 56 through 59 supply base address for 4-word block from Soft Control Memory RAM which is used during each major cycle of multiply operation. Output of the Soft Control Address Bit Generator (ALN 3.1), which increments once each minor cycle, are attached to base address as two low-order bits. Micrand Bit 55 selects one of two addressed 16-bit Soft Control Words. Contents of 12 Soft Control Memory RAM addresses associated with multiply algorithm used in this instruction appear in figure 4-13.

		MINOR CYCLE CONTROL BITS															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
SOFT CONTROL MEMORY ADDRESS (DECIMAL)	00	1	1	1					1					1			
	01	1	1		1	1		1	1					1			
	02	1	1		1	1			1					1			
	03	1	1		1	1			1					1			
	04	1	1		1	1			1					1			
	05	1	1			1			1					1			
	06	1	1						1					1			
	07	1	1						1					1			
	56								1					1			
	57								1				1		1		
	58												1		1		
	59												1		1		

Figure 4-13. Multiply Minor Cycle Control



Soft Control Data Out Bit 7 and inactive Micrand Bit 55 activate Load Enable in Multiply/Divide Minor Cycle Control Register translator (ALN 3.2).

ALN Request activates Select 1 in Multiply/Divide Major Cycle Control Register (ALN 3.2). Select 1 selects Micrand Bits 8 through 11, 16 through 18, 43, 49, 50, 53, 60 and 62. These Major Cycle Control bits direct the multiply operation on 64-ns boundaries.

6. At Time 40, ALN Response partially enables Clear Micrand Register in Micrand Register Control (ALN 3.24).

ALN Response returns to ICP as indication ALN is completing operation specified by micrand.

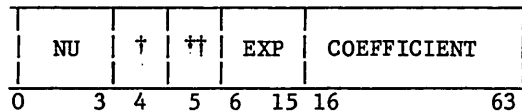
ALN Functional Unit Micrand and decoded control signals enter Control/Micrand Register (ALN 3.0).

Sixty-bit sign-extended floating-point operand from 64-bit  $X_j$  Register in Register File (OPI 3.5) enters B Operand Register (OPI 3.11).

Sixty-bit sign-extended floating-point operand from 64-bit  $X_k$  Register enters C Operand Register (OPI 3.11).

#### NOTE

C170 floating-point operands are in following format.



† Coefficient sign

†† Complemented Exponent sign (bias)

Coefficient is a signed whole number.

Signed exponent is recomplemented if coefficient is negative.

First two soft control bytes enter Multiply/Divide Minor Cycle Control Register (ALN 3.2) to become Minor Cycle Control Bits 0 through 6, 8 through 13. Bits 1 through 3 and 12 are logical ones.

Micrand Bits 8 through 11, 16 through 18, 43, 49, 50, 53, 60, and 62 enter Multiply/Divide Major Cycle Control Register (ALN 3.2), where they remain for 64-ns major cycle. Major Cycle Control Bits 0, 2, 3, 5, 6, and 8 are logical ones.

Second Read Data word enters Soft Control Data Out Register (ALN 3.1).

Soft Control Memory Address Bits 0 through 5 address location 2 in Soft Control Memory RAM (ALN 3.1). Third Read Data word proceeds to Soft Control Data Out Register.

7. Inactive Minor Cycle Control Bit 0 activates Gated Major Cycle Control Bits 5 and 6 (ALN 3.2).
8. C Operand Bits 16 through 63 (multiplier coefficient) are selected in C Register Mux as bits 24 through 71 (ALN 3.13) because Gated Major Cycle Control Bits 5 and 6 are active. Quotient Bits 1 through 16 are selected as bits 8 through 23. Inactive Minor Cycle Control Bits 0 and 8 select C Operand coefficient sign (Quotient Bit 0) in C Register Mux as bits 0 through 7. C Operand Bits 53, 55, 57, 59, 61, and 63 enter C Register Mux as multiplier copy bits 61, 63, 65, 67, 69, and 71.

B Operand Bits 16 through 63 (multiplicand coefficient) enter B Register Mux (ALN 3.14) because Major Cycle Control Bits 2 and 3 are active. B Sign (B Operand coefficient sign) enters mux in bit positions 0 through 15.

Minor Cycle Control Bit 1 activates C Register Clock Enable in C Register Control (ALN 3.13).

Minor Cycle Control Bit 12 selects C Register Data path to return right-shifted C Register Data to the C Register during next minor cycle.

9. ALN Result of first major cycle is not used by instruction. OPI does not write ALN Result into Register File (OPI 3.5) at Time 52.
10. Because Soft Control Data Output Bit 7 is active at Time 41, second pair of soft control bytes enters Multiply/Divide Network Minor Cycle Control Register (ALN 3.2) to become Minor Cycle Control Bits 0 through 6, 8 through 13. Minor Cycle Control Bits 0, 1, 3, 4, 6, and 12 are logical ones.

Third Read Data word enters Soft Control Data Out Register (ALN 3.1).

Soft Control Memory Address Register Bits 0 through 5 address location 3 of Soft Control Memory RAM (ALN 3.1). Fourth Read Data word proceeds to Soft Control Data Out Register.

11. B and C Operands enter B and C Registers, respectively.
12. Multiplicand Bits 0 through 64 and Multiplier Bits 62 through 73 enter Product Network A, B (ALN 3.14).
13. Partial Sum, Carry Bits 0 through 71 are produced in Partial Adder (ALN 3.14). Refer to Multiply/Divide Network, Product Networks A, B for discussion of Partial Sum and Carry bit generation.
14. Minor Cycle Control Bit 0 deactivates Gated Major Cycle Control Bits 5 and 6 (ALN 3.2).
15. Multiply Result Bits 60 through 71 and C Register 12-Bit Right Shift Data Bits 0 through 61 are selected in C Register Muxes (ALN 3.13) by Minor Cycle Control Bit 0.

Minor Cycle Control Bit 1 activates C Register Clock Enable in C Register Control (ALN 3.13).

Minor Cycle Control Bit 12 selects C Register Data path in Multiply/Divide Output Mux (ALN 3.16).

Minor Cycle Control Bit 3 from first minor cycle enters Holding Register (ALN 3.2) to become Minor Cycle Control Bit 3 Delayed.

16. Because Soft Control Data Output Bit 7 is active at Time 42, third pair of selected soft control bytes enters Multiply/Divide Network Minor Cycle Control Register (ALN 3.2) to become Minor Cycle Control Bits 0 through 6, 8 through 13. Minor Cycle Control Bits 0, 1, 3, 4, and 12 activate.

Fourth Read Data word enters Control Data Out Register (ALN 3.1).

17. Minor Cycle Control Bit 3 Delayed (associated with first minor cycle) enters Holding Register in Multiply Partial Summing Network Control (ALN 3.15) and becomes Enable Partial Summing Network Register.

Minor Cycle Control Bits 3 and 4 associated with second minor cycle enter Holding Register (ALN 3.2) to become Minor Cycle Control Bits 3 and 4 Delayed.

Network A, B Partial Sum, Carry Bits 0 through 71 for first minor cycle enter Holding Register (ALN 3.14).

18. Network A Partial Sum Bits 0 through 69, Network A Partial Carry Bits 1 through 67, Network B Partial Sum Bits 0 through 71, and Network B Partial Carry Bits 0 through 69 enter Multiply Partial Summing Network (ALN 3.15).
19. C Register 12-Bit Right Shift Data Bits 0 through 61 and invalid Multiply Result Bits 60 through 71 enter C Register (ALN 3.13). Twelve new bits of multiplier now occupy low-order bit position in C Register.
20. Multiplicand Bits 0 through 64 and new Multiplier Data Bits 62 through 73 enter Product Networks A, B (ALN 3.14) for next partial product cycle.
21. Partial Adder (ALN 3.14) produces second set of Network A, B Partial Sum, Carry Bits 0 through 71.
22. C Register data, right-shifted 12 places for second time and invalid multiply result are selected in C Register Muxes (ALN 3.13) by Minor Cycle Control Bit 0.  
  
Minor Cycle Control Bit 1 activates C Register Clock Enable in C Register Control (ALN 3.13).  
  
Minor Cycle Control Bit 12 selects C Register Data path in Multiply/Divide Output Mux (ALN 3.16).
23. Because Soft Control Data Output Bit 7 is active at Time 43, fourth pair of selected soft control bytes enters Multiply/Divide Network Minor Cycle Control Register (ALN 3.2) to become Minor Cycle Control Bits 0 through 6, 8 through 13. Minor Cycle Control Bits 0, 1, 3, 4, and 12 are active.  
  
Fifth Read Data word, addressed by second ALN Functional Unit Micrand enters Control Data Out Register (ALN 3.1).
24. Multiply Partial Sum, Carry Bits 0 through 75 associated with first minor cycle enter Sum, Carry Register (ALN 3.15).
25. Multiply Partial Carry Bits 5 through 75 and Multiply Partial Sum Bits 4 through 75 enter Multiply Final Adder to produce Multiply Result Bits 0 through 71.
26. Minor Cycle Control Bit 3 Delayed associated with second minor cycle enters Holding Register in Multiply Partial Summing Network Control (ALN 3.15) and becomes Enable Partial Summing Network Register.

Minor Cycle Control Bit 4 Delayed associated with second minor cycle gates first minor cycle's Multiply Partial Sum Bits 0 through 63 to Multiply Partial Summing Network Adder Input (ALN 3.15).

Minor Cycle Control Bits 3 and 4 for third minor cycle enter Holding Register (ALN 3.2) to become Minor Cycle Control Bits 3 and 4 Delayed.

27. Network A, B Partial Sum, Carry Bits 0 through 71 for second minor cycle enter Holding Register (ALN 3.14).

Network A Partial Sum Bits 0 through 69, Network A Partial Carry Bits 1 through 67, Network B Partial Sum Bits 0 through 71 and Network B Partial Carry Bits 0 through 69 for second minor cycle enter Multiply Partial Summing Network (ALN 3.15). Multiply Partial Sum Bits 0 through 63 and Multiply Partial Carry Bits 1 through 64 from previous minor cycle are added to these values.

28. C Register 12-Bit Right Shift Data Bits 0 through 61 and invalid Multiply Result Bits 60 through 71 enter C Register (ALN 3.13).
29. Multiplicand Bits 0 through 64 and Multiplier Data Bits 62 through 73 for third minor cycle enter Product Network A, B (ALN 3.14) for partial multiplication.
30. Third set of Network A, B Partial Sum Carry Bits 0 through 71 is produced in Partial Adder (ALN 3.14).
31. C Register data, right-shifted 12 places for a third time, and valid Multiply Result Bits 60 through 71 in magnitude form are selected in C Register Muxes (ALN 3.13) by Minor Cycle Control Bit 0.

Minor Cycle Control Bit 1 activates C Register Clock Enable in C Register Control (ALN 3.13).

Minor Cycle Control Bit 12 selects C Register Data path in Multiply/Divide Output Mux (ALN 3.16).

32. During second major cycle, pattern established in steps 23 through 31 repeats four times to produce second through fifth 12-bit partial products. Invalid data enters the multiply network in the last two minor cycles of second major cycle, but it is excluded from the final result by soft control.

Major Cycle Control bit values from first ALN Functional Unit Micrand are selected from Holding Register (ALN 3.2) by active Micrand Bit 61 for use during second major cycle. To create fifth 12-bit partial product, second Functional Unit Micrand must address soft control sequence whose first pair of soft control bytes is the same as pair described in step 23. Subsequent soft control acts to terminate calculation once fifth result has been tabulated. (It is natural to assume a 48-bit multiplier will complete a sequence of 12-bit multiplies in four minor cycles. A peculiarity in the design of Product Networks A, B, however, is that the first 12-bit multiplier must be left-shifted one place and the low-order bit position zero-filled for data and parity to align at the output of the Multiply Final Adder (ALN 3.16). A fifth cycle is necessary to include the high-order multiplier bit in the computation.)

At the end of the second major cycle soft control preserves Multiply Partial Sum Bits 4 through 75 and Multiply Partial Carry Bits 5 through 75 in Sum, Carry Registers (ALN 3.15).

During the first minor cycle of the third major cycle the fifth partial product enters the C Register. The first 12-bit partial product enters positions 48 through 59 of the C Register. The C Register latches with the double-precision product (low-order 48 bits) in bit positions 12 through 59.

The contents of the C Register during the eight-minor-cycle 48-bit coefficient multiply appear in figure 4-14. Contents of the C Register at Major Cycle 3, Time 41, are the result of a double-precision multiply operation. During double-precision calculation, the Multiply Final Adder remains latched while the fifth result enters the C Register a second time. The double-precision result right-shifts 12 places into bit positions 24 through 71.

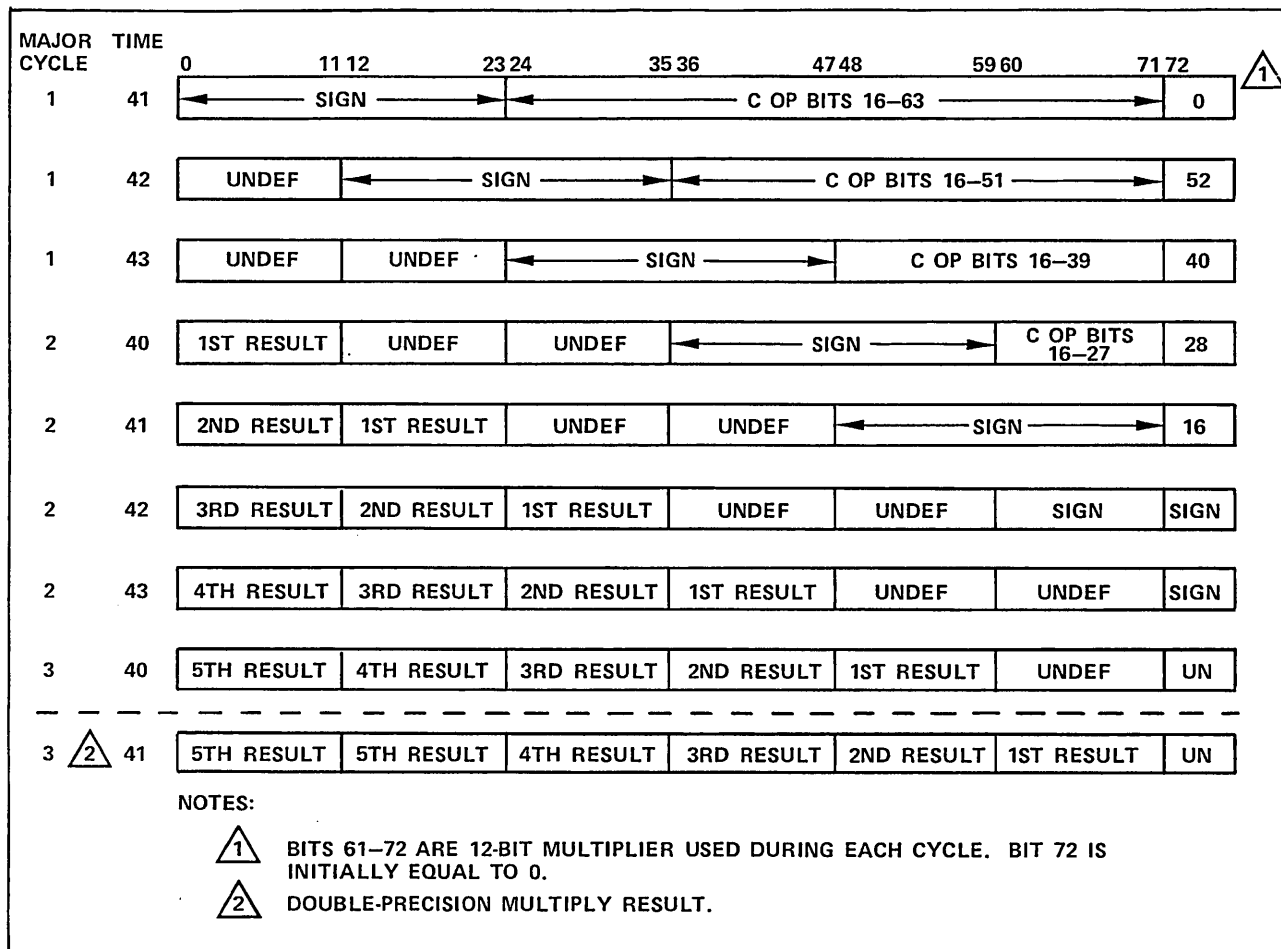


Figure 4-14. C Register Contents

## General Network Operations

1. Refer to C170 Right Shift (21) Instruction, steps 1 and 2, for details of second Functional Unit Micrand's arrival in ALN at Time 31 (Time 41 of first micrand).

Additionally, Micrand Bits 43, 49, 50, 53 through 55 proceed directly to Multiply/Divide Minor and Major Cycle Control Registers (ALN 3.2). Micrand Bits 8 through 11, 16 through 18, 60, and 62 are fanned out from Micrand Decoders (ALN 3.3) to Multiply/Divide Major Cycle Control Register. Micrand Bits 56 through 59 enter Soft Control Memory Address Selector (ALN 3.1). Refer to step 32 in preceding sequence for description of effect Micrand Bits 56 through 59 have on coefficient multiplication.

2. Because Micrand Bits 8 through 11 have a value of  $F_{16}$ , Shift Count Generator Control Bits 4, 8, 11, 16, and 18 activate in Shift Count Generator Field Decoder (ALN 3.3).

Because Micrand Bits 16 through 18 have a value of 0, no Mux Complement Control Bits activate in Complement Field Decoder (ALN 3.3).

Because Micrand Bits 19 through 22 have a value of 7, Exponent Network Control Bits 0, 9, 11, and Exponent Arithmetic Code Equals 7 activate in Exponent Arithmetic Field Decoder (ALN 3.3).

3. All decoded control bits except Exponent Network Control Bit 9 and Exponent Arithmetic Code Equals 7 proceed to Control/Micrand Register input (ALN 3.0).

Exponent Arithmetic Code Equals 7 is enabled at input to Holding Register in Error Result Control (ALN 3.11) by Micrand Bit 23.

Exponent Network Control Bit 9 proceeds to Control/Micrand Register input (ALN 3.0).

4. At Time 32, ALN Request activates in ICP and gates Functional Unit Micrand Bits 12 through 15 to Mux Control Field Decoder (ALN 3.3).

Because Micrand Bits 12 through 15 have a value of  $E_{16}$ , Mux Control Bits 10, 24, 26, and 30 activate, becoming Mux Control Bits 31, 36, 37, and 39.

5. Mux Control bits proceed to Control/Micrand Register (ALN 3.0) input.
6. At Time 33, ALN Go, a 64-ns signal, arrives at ALN Response Control (ALN 3.24) and at data inputs to Control/Micrand Register.
7. At Time 40, (first micrand's Time 50), ALN Response partially enables Clear Micrand Register in Micrand Register Control (ALN 3.24).

ALN Response returns to ICP as indication ALN is completing operation specified by micrand.

Functional Unit Micrand and decoded control signals enter Control Micrand Register (ALN 3.0).

Sixty-bit sign-extended data word from 64-bit  $X_j$  Register in Register File (OPI 3.5) enters B Operand Register (OPI 3.11).

Sixty-bit sign-extended data word from 64-bit  $X_k$  Register enters C Operand Register (OPI 3.11).

Exponent Arithmetic Decode Equals 7 enters Error Result Control Register (ALN 3.11).

8. C Operand Bits 0 through 63/64 through 71 proceed to Shift Network Mux (ALN 3.5).  
 B Operand Bits 1 through 63/64 through 71 proceed to B Register (ALN 3.14) where sign-extended coefficient is selected as multiplicand for ongoing multiplication.  
 C Operand Bits 1 through 15 and B Operand Bits 1 through 15 enter Exponent Arithmetic Network (ALN 3.8).  
 Positive Error Result Enable activates in Error Result Control.
9. Because Mux Control Bits 0 through 15, 17 through 25 are inactive, C Operand Bits 0 through 63 and duplicate bits 64 through 127 are selected in Shift Network Mux (ALN 3.5).
10. Shift Network Mux Bits 0 through 127/128 through 143 pass through Shift Network (ALN 3.6) unshifted. Shift Network Output Bits 80 through 127 (C Operand Bits 16 through 63)/138 through 143 proceed to ALN Output Mux, Bits 16 through 63/Parity (ALN 3.10).
11. Exponent Network Control Bit 0 selects B Operand and C Operand Bits 1 through 15 (packed C170 exponents) to become Mux A and Mux B Bits 1 through 15, respectively, in Exponent Arithmetic Network (ALN 3.8). Exponent Network Control Bit 9 removes exponent biases before unpacked exponents undergo one's complement addition to form product exponent. Unbiased result departs Exponent Arithmetic Network as Exponent Network Result Bits 1 through 15.
12. Exponent Network Result Bits 1 through 15 enter exponent muxes (ALN 3.11). Bits 1 through 7 pass through C170/C180 Exponent Format Selector (ALN 3.11) in C180 format to become Normal Exponent Bits 1 through 7/64 (exponent sign-extended). Normal Exponent Bit 0 is Internal Sign, previously determined coefficient result sign. Bits 8 through 15/65 become Exponent Bits 8 through 15/65 in Normal/Endcase Exponent Selector (ALN 3.11).
13. Shift Network Output Bits 80 through 127/138 through 143 enter ALN Output Mux, Bits 16 through 63/Parity (ALN 3.10) and become ALN Result Bits 16 through 63/66 through 71.  
 Normal Exponent Bits 0 through 7/64 and Exponent Bits 8 through 15/65 become ALN Result Bits 0 through 15/64, 65 is ALN Output Mux 0 through 15/Parity (ALN 3.12).
14. ALN Result Bits 0 through 63/64 through 71 depart for Register File location 08 (OPI 3.5).
15. ALN result enters Register File location 8 at Time 52. ALN result is gated through operand shortstop path in OPI (OPI 3.11) for entry into B and C Operand Registers (OPI 3.11) at third micrand's Time 40.
16. Refer to C170 Right Shift (21) Instruction, steps 1 and 2, for details of third Functional Unit Micrand's arrival in ALN at Time 31 (Time 41 of second micrand).  
 Additionally Micrand Bits 43, 49, 50, 53 through 55 proceed directly to Multiply/Divide Minor and Major Cycle Control Register (ALN 3.2). Micrand Bits 8 through 11, 16 through 18, 60, and 62 are fanned out from Micrand Decoders (ALN 3.3) to Multiply/Divide Major Cycle Control Register. Micrand Bits 56 through 59 enter Soft Control Memory Address Selector (ALN 3.1). Refer to step 32 preceding for description of effect Micrand Bits 56-59 have on coefficient multiplication.
17. Because Micrand Bits 8 through 11 have a value of F16, Shift Count Generator Control Bits 4, 8, 11, 16, and 18 activate in Shift Count Generator Field Decoder (ALN 3.3).

Because Micrand Bits 16 through 18 have a value of 6, Mux Complement Control Bits 0, 1, 5, and 6 activate in Complement Field Decoder (ALN 3.3). Those bits become Complement Control Bits 0 through 3, 7 through 10.

Because Micrand Bits 19 through 22 have a value of 1, Exponent Network Control Bits 1 through 4, 8, 9, and 11 activate in Exponent Arithmetic Field Decoder (ALN 3.3).

18. All decoded control bits proceed to Control/Micrand Register input (ALN 3.0).
19. At Time 32, ALN Request activates in ICP and gates Functional Unit Micrand Bits 12 through 15 to Mux Control Field Decoder (ALN 3.3).  
  
Because Micrand Bits 12 through 15 have a value of 7, Mux Control Bits 0, 1, 10, 24, 26, and 30 activate, and become Mux Control Bits 0 through 2, 31, 36, 37, and 39.
20. Mux Control bits proceed to Control/Micrand Register (ALN 3.0) input.
21. At Time 33, ALN Go, a 64-ns signal, arrives at ALN Response Control (ALN 3.24) and at data inputs to Control/Micrand Register.
22. At Time 40 (second micrand's Time 50), ALN Response partially enables Clear Micrand Register in Micrand Register Control (ALN 3.24).

ALN Response returns to ICP as indication ALN is completing operation specified by micrand.

Shortstopped ALN Result from second micrand enters B and C Operand Registers (OPI 3.11). Because coefficient multiplication has been completed, it is no longer necessary to supply contents of  $X_j$  Register to B Register (ALN 3.14).

23. C Operand Bits 1 through 15 (unpacked sign-extended product exponent) enter Exponent Arithmetic Network (ALN 3.8).
24. If Multiply Result Bit 23 is active, indicating coefficient overflow, General Network Right Shift One activates in Multiply/Divide Right Shift One Translator (ALN 3.18). Normalizing algorithm right-shifts unnormalized coefficient one place. Both Operands Normalized must also be active to generate signal. Both Operands Normalized results from test performed on input operands' most significant coefficient bits and coefficient signs during first major cycle.
25. Micrand Bits 23 through 25, with decoded value of three, activate Force Count to  $OF_{16}$  or Force Count to  $OE_{16}$  in Normalize Encoder (ALN 3.7), depending on state of General Network Right Shift One.
26. Significance Encoder Bits 0 through 5 (containing value of  $OE_{16}$  or  $OF_{16}$ ) enters Significance Count Adjustment Adder (ALN 3.7) in complemented form. Bits are added to Micrand Bits 0 through 6 (containing value  $3E_{16}$ ) and Add One to Exponent. Because single-precision coefficient reflects 48 high-order bits of 96-bit result, 48 places must be added to product exponent when coefficient does not need to be adjusted for unnormalized result. Forty-nine places must be added to exponent if product coefficient must be right-shifted one place. Because low-order multiplier bit is a zero, the product exponent must also be adjusted by minus one. Making least significant multiplier bit a zero effectively left-shifts product coefficient one place. Normalize Count Bits 0 through 7 depart Significance Count Adjustment Adder containing values  $30_{16}$  ( $48_{10}$ ) or  $2F_{16}$  ( $47_{10}$ ).



27. Because Exponent Network Control Bits 1 and 4 are active Normalize Count Bits 1 through 15 and C Operand Bits 1 through 15 become Mux A and B Bits 1 through 15 respectively, in Exponent Arithmetic Network (ALN 3.8). Exponent Network Control Bits 8, 9 pass Mux A and B bits unbiased to inputs of Adder. Adder Output Bits 1 through 15 are placed in C170 biased format in Output Control and become Exponent Network Result Bits 1 through 15.
28. Exponent Network Result Bits 1 through 15 enter exponent muxes (ALN 3.11). Exponent Network Result Bits 8 through 15/65 become Exponent Bits 8 through 15/65 in Normal/Endcase Exponent Selector (ALN 3.11).  
  
Exponent Network Result Bits 5 through 7 and C Operand Sign, the preserved Gated Multiply/Divide Result Sign from first micrand, are selected by Exponent Network Control Bit 2 as C170 format exponent, in C170/C180 Exponent Format Selector (ALN 3.11). Normal Exponent Bits 0 through 7/64 and Exponent Bits 8 through 15/65 proceed to ALN Output Mux, Bits 0 through 15/Parity (ALN 3.12).
29. At Time 41 of third micrand, Minor Cycle Control Bits 11 and 13 select Multiply Result Bits 8 through 71/Parity in Multiply/Divide Output Mux (ALN 3.16). Single-precision product coefficient occupies bits positions 24 through 71.
30. B Operand or Multiply/Divide Result Bits 0 through 63/64 through 71 proceed to Shift Network Mux (ALN. 3.5).
31. Because Complement Control Bits 0 through 3 and 7 through 10 are active, B Operand Bits 0 through 127 are complemented if B Operand Sign is positive (B Operand Bit 0, calculated product coefficient sign, returning from OPI). Refer to step 12 preceding. Because Shift Network Output data is complemented entering Large Adder, complementing positive coefficient or not complementing negative coefficient here ultimately converts it to signed operand form in Large Adder.
32. Because Mux Control Bits 0 through 2 are active, B Operand Bits 0 through 63 become Shift Network Mux Bits 0 through 63/128 through 135 in Shift Network Mux.
33. Significance Control  $OE_{16}$  or  $OF_{16}$  enters Shift Network (ALN 3.6) in complemented form to right-shift Shift Network Mux Bits 0 through 63/128 through 135, 48 or 49 places, depending on whether overflow correction is required.
34. Shift Network Output Bits 16 through 111/130 through 141 enter Large Adder with normalized coefficient in bit positions 64 through 111. Micrand Bits 26 through 32 enable 64-bit two's complement addition of Shift Network Output Bits 16 through 111 and zeros in Adder Mux Bits 0 through 95.
35. Micrand Bit 37 selects Large Adder Result Bits 48 through 95/102 through 107 to become ALN Result Bits 16 through 63/66 through 71 (ALN 3.10). Bits 48 through 95 are normalized, single-precision, floating-point product coefficient is signed form.  
  
The double-precision product coefficient occupies bit positions 12 through 59 in C Register (ALN 3.13). This portion of the 96-bit product coefficient is not used by this instruction.  
  
Micrand Bit 34 selects Normal Exponent Bits 0 through 7/64 and Exponent Bits 8 through 15/65 to become ALN Result Bits 0 through 15/64, 65 (ALN 3.12). These bits are biased C170 format floating-point product exponent and coefficient sign.
36. ALN Result Bits 0 through 15/64, 65 and 16 through 63/66 through 71 proceed to  $X_i$  Register is Register File (OPI 3.5).

37. Clear Micrand Register activates in Micrand Register Control (ALN 3.24) if ALN Go deactivates in ICP at third micrand's Time 43. Inactive ALN Go signifies next micrand is not being issued to ALN.
38. Clear Micrand Register clears Control/Micrand Register (ALN 3.0) before deactivating at Time 50.
39. Result enters  $X_i$  Register at Time 52.

#### C180 INTEGER QUOTIENT (27) INSTRUCTION

The instruction divides the 64-bit word in the  $X_k$  Register of the Register File (OPI 3.5) by the 64-bit word in the  $X_j$  Register. The 64-bit quotient enters the  $X_k$  Register.

The instruction issues four ALN Functional Unit Micrands, with values of 01FE E000 0000 02B8<sub>16</sub>, 01FE E000 0000 009C<sub>16</sub>, 010E 0000 00C0 0094<sub>16</sub>, and 6387 C017 5460 00AC<sub>16</sub>. The second micrand is issued to ALN 15 consecutive times.

During the first major cycle the contents of the  $X_k$  Register become the B Operand. The contents of the  $X_j$  Register become the C Operand. The C Operand enters the divisor register in the Divide Network (ALN 1.0). The B Operand dividend enters the quotient register in the Divide Network. The Divide Network tests for a zero divisor (divide fault).

During the next 16 major cycles, 64 quotient bits are generated in sequence (from most significant to least significant bit) in the Divide Network. The quotient register supplies a dividend bit to the dividend register at the rate of one bit per minor cycle, and receives a quotient bit at the same rate from an adder/subtractor network.

In the final major cycle, the integer quotient contained in the quotient register enters the C Register (ALN 1.0). C Register Data proceeds to the Multiply/Divide Output Mux. The Shift Network Mux selects B Operand or Multiply/Divide Result and the Shifter Ranks 1, 2 shift the quotient into proper position for entry into the Large Adder.

The quotient proceeds unmodified through the Large Adder to the ALN Output Mux (ALN 1.0). ALN Result enters  $X_k$  Register in the Register File.

1. Refer to C170 Right Shift (21) Instruction, steps 1 and 2, for details of first ALN Functional Unit Micrand's arrival in ALN at micrand sequence Time 31.

Additionally, Micrand Bits 43, 49, 50, 53 through 55, and 61 proceed directly to Multiply/Divide Minor and Major Cycle Control Registers (ALN 3.2). Micrand Bits 8 through 11, 16 through 18, 60, and 62 are fanned out from Micrand Decoders (ALN 3.3) to Multiply/Divide Major Cycle Control Register. Micrand Bits 56 through 59 enter Soft Control Memory Address Selector (ALN 3.1).

2. Micrand Bits 8 through 11, having a value of  $F_{16}$ , are used in Multiply/Divide Network. Activated Shift Count Generator Field Decoder (ALN 3.2) bits are invalid. Micrand Bits 16 through 18, similarly, are employed only in Multiply/Divide Network.
3. At Time 32, Soft Control Address Bits 0 through 5 (with a value of 44<sub>10</sub>) enter Soft Control Memory Address Register (ALN 3.1). Soft Control Address Bits 4, 5 from Soft Control Address Bit Generator (ALN 3.1) are equal to zero during PH2.

Inactive Micrand Bit 55 enters Holding Register in Multiply/Divide Minor Cycle Control Register (ALN 3.2).

4. Soft Control Memory Address Bits 0 through 5 address location 44<sub>10</sub> in Soft Control Memory RAM (ALN 3.1). Read Data Bits 0 through 31/32 through 35 proceed to Soft Control Data Out Register (ALN 3.1).
5. At Time 33, ALN Go, a 64-ns signal, arrives at ALN Response Control (ALN 3.24) and data inputs to Control/Micrand Register (ALN 3.0).

Read Data Bits 0 through 31/32 through 35 enter Soft Control Data Out Register (ALN 3.1) to become Soft Control Data Out Bits 0 through 31/32 through 35.

Soft Control Memory Address Bits 0 through 5 address location 45<sub>10</sub> of Soft Control Memory RAM (ALN 3.1). Second Read Data word proceeds to Soft Control Data Out Register.

Inactive Micrand Bit 55 enters 16-ns Delay in Multiply/Divide Minor Cycle Control Register (ALN 3.2).

ALN Request enters Holding Register (ALN 3.3).

6. ALN Request and inactive Micrand Bit 55 select Soft Control Data Out Bits 0 through 15/32, 33 in Multiply/Divide Minor Cycle Control Register.

The 16 selected bits provide soft control for first minor cycle of divide operation. Soft Control bit assignments are shown in figure 4-13.

#### NOTE

Micrand Bits 56 through 59 supply base address for a 4-word block from Soft Control Memory RAM which is used during each major cycle of divide operations. Output of Soft Control Address Bit Generator (ALN 3.1), which increments once each minor cycle, joins base address and becomes two low-order bits of RAM address. Micrand Bit 55 selects one of two 16-bit Soft Control words from each RAM address. Contents of 12 Soft Control Memory RAM addresses associated with divide algorithm appear in figure 4-15.

		MINOR CYCLE CONTROL BITS															
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
SOFT CONTROL MEMORY ADDRESS (DECIMAL)	44	1	1	1	1				1			1					
	45							1	1			1					
	46					1	1		1			1					
	47					1	1		1			1					
	36					1	1		1		1	1					
	37					1	1		1			1					
	38					1	1		1		1	1					
	39					1	1		1			1					
	40	1							1						1		
	41								1						1		
	42														1		
	43														1		

Figure 4-15. Integer Divide Soft Control

Soft Control Data Out Bit 7 and inactive Micrand Bit 55 activate Load Enable in Multiply/Divide Minor Cycle Control Register translator (ALN 3.2).

ALN Request activates Select 1 in Multiply/Divide Major Cycle Control Register (ALN 3.2). Select 1 selects Micrand Bits 8 through 11, 16 through 18, 43, 49, 50, 53, 60, and 62. These Major Cycle Control bits provide control of divide operations on 64-ns boundaries.

7. At Time 40, ALN Response partially enables Clear Micrand Register in Micrand Register Control (ALN 3.24).

ALN Response returns to ICP as indication ALN is completing operation specified by micrand.

ALN Functional Unit Micrand enters Control Micrand Register (ALN 3.0).

Sixty-four bit signed integer dividend from  $X_k$  Register enters B Operand Register (OPI 3.11).

Sixty-four bit signed integer divisor from  $X_j$  Register enters C Operand Register (OPI 3.11).

First two Soft Control bytes enter Multiply/Divide Minor Cycle Control Register (ALN 3.2) to become Minor Cycle Control Bits 0 through 6, and 8 through 13. Bits 1 through 4, and 10 are logical ones.

Micrand Bits 8 through 11, 16 through 18, 43, 49, 50, 53, 60, and 62 enter Multiply/Divide Major Cycle Control Register (ALN 3.2), where they remain for 64-ns major cycle. Major Cycle Control Bits 0 through 7 are logical ones.

Second Read Data word enters Soft Control Data Out Register (ALN 3.1). Soft Control Memory Address Bits 0 through 5 address location 46<sub>10</sub> of Soft Control Memory RAM (ALN 3.1). Third Read Data word proceeds to Soft Control Data Out Register.

8. Major Cycle Control Bit 5 selects C Operand Bit 0 to become C Negative (ALN 3.13).

Major Cycle Control Bit 2 selects B Operand Bit 0 to become B Negative in B Operand Sign Selector (ALN 3.14).

Major Cycle Control Bit 4 selects C Operand Bit 0 to become Selected C Sign in C Sign Selector (ALN 3.13).

Minor Cycle Control Bit 3 activates B Sign Gated in B Operand Sign Selector (ALN 3.14).

9. Minor Cycle Control Bit 3 selects B Operand Bits 1 through 63 (dividend) and B Sign Gated at input of Quotient Register (ALN 3.17).

Minor Cycle Control Bit 10 partially enables Quotient and Sign registers load term (ALN 3.17) and Holding Register load term in Iteration Result Network (ALN 3.17).

Major Cycle Control Bits 1 or 13, and 2 enable C Operand Bits 1 through 63 (divisor) and Selected C Sign to input of Divisor Register (ALN 3.17). Selected C Sign also is gated to input of Divisor Sign Register (ALN 3.17).

10. ALN Result for first micrand is not used by instruction. Inactive Mux Control Bits 0 through 15, 17 through 25 from Control/Micrand Register (ALN 3.0) select C Operand from OPI in Shift Network Mux (ALN 3.5). General network passes it shifted through Shift Network to ALN Output Muxes, Bits 0 through 15, 16 through 63/Parity (ALN 3.10, 3.12) and OPI. OPI does not write ALN Result into Register File (OPI 3.5).

11. At Time 41, second pair of soft control bytes enters Multiply/Divide Network Minor Cycle Control Register (ALN 3.2) to become Minor Cycle Control Bits 0 through 6, 8 through 13. Minor Cycle Control Bits 6, 7, and 10 are logical ones.

Third Read Data word enters Control Data Out Register (ALN 3.1). Soft Control Memory Address Register Bits 0 through 5 address location 47<sub>10</sub> of Soft Control Memory RAM (ALN 3.1). Fourth Read Data word proceeds to Soft Control Data Out Register.

12. B Sign Gated enters Sign Register (ALN 3.17).

B Operand Bits 1 through 63 and B Sign Gated enter Quotient Register (ALN 3.1).

Selected C Sign enters Divisor Sign Register (ALN 3.17).

C Operand Bits 1 through 63 and Selected C Sign enter Divisor Register (ALN 3.17).

Inactive Minor Cycle Control Bit 5 enters Holding Register in Iteration Result Network (ALN 3.17).

13. B Sign Gated deactivates in B Operand Sign Selector (ALN 3.14).

Minor Cycle Control Bit 6 partially enables Holding Register load term in B Operand Sign Selector (ALN 3.14) and load term at Holding Register (ALN 3.2).

Minor Cycle Control Bit 10 partially enables Holding Register load term in Iteration Result Network.

14. If B Sign Gated indicates B Operand is negative value, Quotient Bits 0 through 63 are complemented in One's Complement Control (ALN 3.17).

Because negative C180 signed operand is represented in two's complement form, one is added to one's complemented value. Quotient Bits 0 through 63 are magnitude form of integer dividend.

If selected C Sign indicates divisor is negative, C Operand Bits 0 through 63 are complemented at output of Divisor Register (ALN 3.17). Because C180 signed operand is represented in two's complement form, two's complement plus one adjustment is supplied from Holding Register (ALN 3.17) to adder/subtractor in Divide Network to create magnitude-form divisor.

15. Inactive Minor Cycle Control Bit 3 enables Quotient Bits 1 through 63, left-shifted one place, to Quotient Register input. Completed Divide Result (Bit 63) is a logical zero because Minor Cycle Control Bit 10 Delayed is inactive.

16. At Time 42, third pair of Soft Control bytes enters Multiply/Divide Network Minor Cycle Control Register (ALN 3.2) to become Minor Cycle Control Bits 0 through 6, 8 through 13. Minor Cycle Control Bits 4, 5, 7, and 10 are logical ones.

Fourth Read Data word enters Control Data Out Register (ALN 3.1).

17. Quotient Bit 0 (high-order dividend bit) enters Holding Register (ALN 3.17).

Shifted Quotient Bits 0 through 62 enter Quotient Register.

Sign Register (ALN 3.17) clears because B Sign Gated is inactive.

Dividend Register and Carry Register (ALN 3.17) clear because Minor Cycle Control Bits 2 and 4 of previous cycle were inactive.

Selected C Sign and C Operand Bits 1 through 63 reenter Divisor Sign and Divisor Register and two's complementing of divisor recurs as required.

Minor Cycle Control Bit 10 enters Holding Register (ALN 3.2).

Multiply Divide Result Sign, created from B Negative and C Negative, enters Holding Register in B Operand Sign Selector (ALN 3.14) and becomes Gated Multiply Divide Result Sign.

Inactive Minor Cycle Control Bit 5 enters Holding Register in Iteration Result Network (ALN 3.17).

18. Uncomplemented Quotient Bits 0 through 63 proceed unincremented to Divide Network output, because Minor Cycle Control Bit 4 is active and Bit 3 inactive. Quotient Bit 0 (second dividend bit) proceeds to Holding Register (ALN 3.17). Left-Shifted Quotient Bits 1 through 63 proceed to Quotient Register.

19. Inactive Minor Cycle Control Bit 5 activates Select Subtract Results in Iteration Result Network (ALN 3.17).
20. Remainder and carry bits from subtraction proceed through left shifter to Dividend and Carry Register inputs because Minor Cycle Control Bit 4 is active.  
  
Remainder Register Least Significant Input Bit activates with arrival of latched Quotient Bit 0 in Division Remainder Least Significant Bit Selector (ALN 3.17).
21. Remainder Register Least Significant Input Bit, high-order dividend bit, proceeds to Dividend Register input in low-order bit position.
22. Group Generate bits and Group Enable bit proceed to input of Holding Register in Iteration Result Network, where Minor Cycle Control Bit 10 partially enables load term.  
  
Active Minor Cycle Control Bit 5 proceeds to Iteration Result Network Holding Register input.
23. At Time 43, fourth pair of selected soft control bytes enters Multiply/Divide Network Minor Cycle Control Register (ALN 3.2) to become Minor Cycle Control Bits 0 through 6, 8 through 13, Minor Cycle Control Bits 4, 5, 7, and 10 are logical ones.  
  
Fifth Read Data word, addressed by second micrand, enters Control Data Out Register (ALN 3.1).
24. Quotient Bit 0 (second dividend bit) enters Holding Register (ALN 3.17).  
  
Shifted Quotient Bits 0 through 62 enter Quotient Register.  
  
Sign Register (ALN 3.17) remains clear because B Sign Gated is inactive.  
  
Selected C Sign and C Operand Bits 1 through 63 reenter Divisor Sign and Divisor Registers and two's complementing of divisor recurs as required.  
  
Active Minor Cycle Control Bit 5 enters Holding Register in Iteration Result Network (ALN 3.17) along with Group Generate Bits 1 through 16 and Group Enable Bits 0 through 15 from forced subtraction of previous cycle.  
  
Remainder In Bits 0 through 66 from forced subtraction, Remainder Least Significant Input Bit (high-order dividend bit) and Carry In Bits 0 through 16 from forced subtraction enter Dividend and Carry Registers (ALN 3.17).
25. Latched Quotient Bit 0 (second dividend bit) becomes Remainder Register Least Significant Bit in Divisor Remainder Least Significant Bit Selector (ALN 3.17). The bit proceeds to Dividend Register input.  
  
Uncomplemented New Quotient Bits 0 through 63 proceed unincremented to Divide Network Output because Minor Cycle Control Bit 4 is active and Bit 3 is inactive. Quotient Bit 0 (third dividend bit) proceeds to input of Holding Register (ALN 3.17). Left-shifted Quotient Bits 1 through 63 proceed to Quotient Register input.  
  
Contents of Dividend and Carry Registers are added to produce left-shifted, carry-adjusted remainder of forced subtraction.

Group Generate Bits 1 through 16 and Group Enable Bits 0 through 15 produce Carry Out (equal to zero if divide fault condition does not exist) in Iteration Result Network. Carry Out serves as divide fault test bit in first integer divide cycle. Inactive Carry Out enables sum of divisor and two times the remainder to be output from adder/subtractor network. For details of restoring divide algorithm, refer to Multiply/Divide Network, Division discussion.

The new remainder (Remainder Out Bits 1 through 67 and Carry Out Bits 1 through 16) proceeds to Dividend and Carry Registers.

Completed Divide Result proceeds to Quotient Register input as least significant bit.

26. Active Minor Cycle Control Bit 5 proceeds to Iteration Result Network Holding Register input.

Group Generate bits and Group Enable bits proceed to input of Holding Register of Iteration Result Network, where Minor Cycle Control Bit 10 partially enables load term. These bits produce first quotient bit at Time 50.

The sequence outlined in steps 23 through 26 repeats with minor variations to introduce a new dividend bit into the Dividend Register and store a new quotient bit in the Quotient Register each minor cycle until 64-bit magnitude-form quotient has been generated.

The Instruction Unit micrand sequence enters a loop which results in Soft Control Memory RAM (ALN 3.1) locations 36 through 39<sub>10</sub> being addressed 16 consecutive times. Locations 36 through 39<sub>10</sub> contain active Minor Cycle Control Bits 4, 5, 7, and 10.

In addition to addressing RAM locations 36 through 39<sub>10</sub> for the final time, the seventeenth Functional Unit Micrand issued enters zeros in the Holding Register at the input of the Multiply/Divide Major Cycle Control Register Mux (ALN 3.2). These bits become the Major Cycle Control bits for the final major cycle of the instruction.

#### Quotient Storage

1. Refer to C170 Right Shift (21) Instruction, steps 1 and 2, for details of eighteenth micrand's arrival in ALN at Time 31.

Micrand Bits 56 through 59 enter Soft Control Memory Address Selector (ALN 3.1).

2. Because Micrand Bits 16 through 18 have a value of six, Mux Complement Control Bits 0, 1, 5, and 6 activate in Complement Field Decoder (ALN 3.3), becoming Complement Control Bits 0 through 3, and 7 through 10.
3. All decoded control bits proceed to Control/Micrand Register input (ALN 3.0).
4. At Time 32, ALN Request activates in ICP and gates Micrand Bits 12 through 15 to Mux Control Field Decoder (ALN 3.3).

Because Micrand Bits 12 through 15 have a value of seven, Mux Control Bits 0, 1, 10, 24, 26, and 30 activate, becoming Mux Control Bits 0, 1, 2, 31, 36, 37, and 39.

Soft Control Address Bits 0 through 5 (with a value of 40<sub>10</sub>) enter Soft Control Memory Address Register (ALN 3.1).



Inactive Micrand Bit 55 enters Holding Register in Multiply/Divide Minor Cycle Control Register (ALN 3.2).

5. Mux Control Bits proceed to Control/Micrand Register input (ALN 3.0).

Soft Control Memory Address Bits 0 through 5 address location 40<sub>10</sub> in Soft Control Memory RAM (ALN 3.1). Read Data Bits 0 through 31/32 through 35 proceed to Soft Control Data Out Register (ALN 3.1).

6. At Time 33, ALN Go arrives at ALN Response Control (ALN 3.24) and data inputs to Control/Micrand Register (ALN 3.0).

For description of loading of Major and Minor Cycle Control bits, refer to steps 5 and 6 of preceding sequence associated with integer divide instruction.

7. At Time 40, ALN Response partially enables Clear Micrand Register in Micrand Register Control (ALN 3.24).

ALN Response returns to ICP as indication ALN is completing operation specified by micrand.

ALN Functional Unit Micrand and decoded control signals enter Control/Micrand Register (ALN 3.0).

First two Soft Control bytes enter Multiply/Divide Minor Cycle Control Register (ALN 3.2) to become Minor Cycle Control Bits 0 through 6, 8 through 13. Bits 1, 7, and 12 are logical ones. Bit 1 remains active for one minor cycle, bit 12 for four minor cycles.

No Major Cycle Control Bits activate.

8. Minor Cycle Control Bit 1 activates C Register Clock Enable in C Register Control (ALN 3.13).

Minor Cycle Control Bit 12 selects C Register input to Multiply/Divide Output Mux (ALN 3.16).

Quotient Bits 1 through 63 and Completed Divide Result are selected in C Register input mux (ALN 3.13).

9. At Time 41, Quotient Bits 1 through 63 are latched in C Register (ALN 3.13) as C Register Data Bits 8 through 70.
10. C Register Data Bits 8 through 71 are selected in Multiply/Divide Output Mux (ALN 3.16) for remainder of last major cycle.
11. B Operand or Multiply Divide Result Bits 0 through 63/64 through 71 proceed to Shift Network Mux (ALN 3.5) where Complement Control Bits 0 through 3, 7 through 10 complement Quotient if B Operand Sign is positive. B Operand Sign is Gated Multiply Divide Result Sign, computed in step 17 of previous sequence. Because Shift Network (ALN 3.16) Output is complemented entering Large Adder, selective complementing in Shift Network Mux ultimately results in signed operands being created in Large Adder from magnitude form values.
12. Mux Control Bits 0 through 2 select Quotient as Shift Network Mux Bits 0 through 63/128 through 135.

13. Shift Network Mux Bits 0 through 127/128 through 143 proceed to Shift Network.
14. Constant of  $49_{10}$  contained in Micrand Bits 0 through 6 of final micrand right shifts Quotient to bit positions 49 through 111 in Shifter Ranks 1, 2. Integer Multiply Divide Overflow Bit, which occupies bit position 48, is Quotient Bit 0 selected in C Register (ALN 3.13) to be C Register Data Bit 7.
15. Shift Network Output Bits 16 through 111/130 through 141 proceed to Large Adder (ALN 3.10).  
  
Adder Mux Bits 0 through 95/96 through 107 (equal to zero) are present at Large Adder.
16. Large Adder ALU performs two's complement add of two inputs. Signed integer quotient departs Large Adder as Large Adder Result Bits 32 through 95/100 through 107. Micrand Bit 33 provides plus one input to create two's complement quotient if C Operand Sign is negative.
17. Micrand Bit 37 selects Large Adder Result Bits 48 through 95/102 through 107 in ALN Output Mux, Bits 16 through 63/Parity (ALN 3.10).  
  
Micrand Bit 35 selects Large Adder Result Bits 32 through 47/100, 101 in ALN Output Mux, Bits 0 through 15/Parity (ALN 3.12).
18. ALN Result Bits 0 through 63/64 through 71 proceed to  $X_k$  Register in Register File (OPI 3.5).
19. Clear Micrand Register activates in Micrand Register Control (ALN 3.24) if ALN Go deactivates in ICP at next micrand's Time 33. Inactive ALN Go signifies next micrand is not being issued to ALN.
20. Clear Micrand Register clears Control/Micrand Register (ALN 3.0) before deactivating at Time 50.
21. ALN result enters  $X_k$  Register at Time 52.

#### IMPLICIT COPY OPERATION

If the CP issues a Functional Unit Micrand to a functional unit other than ALN, ALN provides a path for the C Operand to pass through ALN unmodified. Typically, the General Micrand uses this implicit copy path to transfer an operand from one Register File (OPI 1.0) location to another while the Functional Unit Micrand is executing in AC or BDP.

The C Operand enters the C Register (ALN 1.0) at micrand execution Time 41 and proceeds to the Multiply/Divide Output Mux. B Operand or Multiply/Divide Result enters the Shift Network Mux and passes through Shifter Ranks 1 and 2 to the ALN Output Mux (Lower and Upper). The implicit copy operand returns to OPI as write data. A copy of the C Operand always passes through ALN during major cycles in which ALN Go is inactive. Whether the implicit copy enters the Register File is determined by the General Micrand.

The following implicit copy sequence description begins at Time 42 of a Functional Unit Micrand executing in ALN. The next micrand in the pipeline (which is not issuing a Functional Unit Micrand to ALN) is at Time 32 in its execution cycle.

1. At Time 32 ALN Request does not activate in Functional Unit Go Control (ICP 3.5).  
  
Functional Unit Code Equals 0, 1 do not activate in General Micrand Functional Unit Field Decoder (ICP 3.5).

2. Inactive ALN Request activates Mux Control Bit 2 in Mux Control Field Decoder (ALN 3.3).

Inactive ALN Request disables Functional Unit Micrand Bit 61 (ALN 3.3) and Functional Unit Micrand Bit 54 in Multiply/Divide Major Cycle Control Register (ALN 3.2). It also partially enables Load Enable and selects hardwired value 0008<sub>16</sub> in Multiply/Divide Minor Cycle Control Register (ALN 3.2).

ALN Go does not activate in ALN, BDP Go Control (ICP 3.5).

3. Mux Control Bit 2 becomes Mux Control Bits 3, 4, and 6 (ALN 3.3).

Inactive Functional Unit Micrand Bits 54 and 61 select hardwired FE00<sub>16</sub> in Multiply/Divide Minor Cycle Control Register Mux (ALN 3.2).

4. At Time 33, ALN Go deactivates in ALN, BDP Go Control Holding Register (ICP 3.5).
5. At Time 40, active ALN Response (from previous micrand) and inactive ALN Go load Mux Control Bits 0 through 11, 26 through 31, and 38 into Control Register (ALN 3.0).

ALN Response deactivates in ALN Response Control (ALN 3.24). Control Register remains latched until next time ICP issues ALN Go.

FE00<sub>16</sub> enters Multiply/Divide Major Cycle Control Register, causing Major Cycle Control Bits 0 through 6 to activate.

0008<sub>16</sub> enters Multiply/Divide Minor Cycle Control Register, causing Minor Cycle Control Bit 12 to activate.

C Operand Bits 0 through 63/64 through 71 arrive at C Register input (ALN 3.13).

Set Rank 40 Valid and inactive ALN Request enter Holding Register in C Register Control (ALN 3.13), activating C Register Clock Enable.

6. Mux Control Bits 3, 4, and 6 enable selection of C Operand in bit positions 0 through 63 and B Operand in bit positions 64 through 127 of Shift Network Mux (ALN 3.5).

Major Cycle Control Bits 4 through 6 become Gated Major Cycle Control Bits 4 or 13, 5, and 6 (ALN 3.2) and select C Operand Bits 0 through 63 in C Register Mux (ALN 3.13).

Minor Cycle Control Bit 12 enables selection of C Register Data Bits 8 through 71 in Multiply/Divide Output Mux (ALN 3.16).

7. At Time 41, C Operand Bits 0 through 63 enter C Register and become C Register Data Bits 8 through 71.
8. C Register Data Bits 8 through 71 enter Parity Generator (ALN 3.13) to produce C Register Parity 0 through 7.
9. C Register Data Bits 8 through 71 and C Register Parity 0 through 7 enter Multiply/Divide Output Mux (ALN 3.16), becoming B Operand or Multiply/Divide Result Bits 0 through 63/64 through 71.
10. B Operand or Multiply/Divide Result Bits 0 through 63/64 through 71 enter Shift Network Mux (ALN 3.5) and are fanned out to bit positions 0 through 127/128 through 143.

11. Shift Network Mux selects 64-bit implicit copy operand to become Shift Network Mux Bits 64 through 127/136 through 143.
12. Shift Network Mux Bits 64 through 127/136 through 143 pass through Shift Network unshifted, becoming Shift Network Output Bits 64 through 127/136 through 143.
13. Shift Network Output Bits 80 through 127/138 through 143 enter ALN Output Mux, Bits 16 through 63/Parity (ALN 3.10), where they become ALN Result Bits 16 through 63/66 through 71.

Shift Network Output Bits 64 through 79/136, 137 proceed to ALN Output Mux, Bits 0 through 15/Parity (ALN 3.12), where they become ALN Result Bits 0 through 15/64, 65.

14. Implicit copy operand returns to OPI as write data.

PART 2

ADDRESS CONTROL (AC)



This part of this section describes AC which performs three major functions under micrand and soft control.

- Forms byte number of AC Address sent to LM for Register File data or BDP stream data.
- Provides assembly/disassembly data paths and control for A and B streams from LM to BDP and C stream from BDP to LM.
- Processes load/store instructions.

Refer to AC 1.0 for Soft Control, Address Formation, A/C Stream Disassembly/Assembly, B Stream Disassembly, BDP Instructions, Load/Store Instructions, and Test and Set Bit Instruction descriptions. Refer to AC 3.0 through 18 for AC Sequences.

#### MICRAND CONTROL

CST sends the AC micrand field to AC to provide major cycle control. This 17-bit field is part of the AC micrand which is described in section 3.

#### SOFT CONTROL

Soft Control 1 through 3 (AC 1.0) provide minor cycle control for the various AC Sequences. Soft Control 1 and 2 are RAMs which control A/C stream load/store operations and are loaded from the MAC Data Bus during system initialization. Refer to Soft Control Memory Write in section 2. Pseudo Soft Control 3 is a hardware decoder which controls B stream load operations. Micrand Bits 0 through 3 point to the first soft control address of an AC sequence. Once a sequence is entered, the lower (rightmost) six bits of each soft control location, together with various control signals, point to the next control word. Refer to Maintenance Aids in the Maintenance and Parts Data manual for maps of AC Soft Control Memories.

#### SOFT CONTROL 1

This 32-bit 64-word RAM provides minor cycle address control for the A and C streams.

<u>Bit</u>	<u>Description</u>
0	Enable feedback path to Address Adder input A (AC 3.14).
1	Block A/C Stream Address Count to LM Byte Address Select Mux (AC 3.15). Enable Address Adder to A/C and B Stream Address Counters (AC 3.15).
2	Enable Load/Store Multiple Length Adder to Address Adder through Address Offset Select Mux (AC 3.13). Block A/B Operand to Address Adder input B (AC 3.14).
3	Load A/C Stream Length Counter (AC 3.13).
4	Enable A Stream Memory Request (AC 3.5). Enable C Stream Memory Request (AC 3.6). Enable set C1 Buffer Full FF (AC 3.6).
5	Block A/B Operand to Address Adder input B (AC 3.14).

<u>Bit</u>	<u>Description</u>
6	Enable Soft Control 2 address change (AC 3.2).
7	Not used.
8	Enable input to Load/Store Multiple Length Adder (AC 3.13). Enable address restriction on load/store multiple operations (AC 3.18).
9	Not used.
10	Load Byte Shift Counter (AC 3.9).
11	Enable decrement A/C Stream Length Counter (AC 3.13).
12	Enable read Soft Control 2 (AC 3.2).
13	Load A/C Disassembly/Assembly Counter (AC 3.9).
14	Enable set C Stream Operation FF (AC 3.0). Load Final A/C Stream Byte Number Register (AC 3.9).
15	Enable Address Adder to Mark Bit Generator through Mark Mask Mux (AC 3.17).
16	Complement Byte Shift Count (AC 3.14).
17	Enable set A Stream End FF (AC 3.7).
18	Enable read Soft Control 1 during AC no operation (AC 3.1).
19	Enable set Test Address FF (AC 3.4).
20	Send Load Aj Descriptor signal to BDP (AC 3.7).
21	Send AC Response signal to ICP and LM (AC 3.4).
22	Decrement B Stream Disassembly Counter (AC 3.12).
23	Enable read Soft Control 1 (AC 3.1). Hold AC A Stream Busy (AC 3.4).
24	Enable set Page Boundary FF (AC 3.16).
25	Load Bit Right Shift Counter (AC 3.12).
26-31	Complement of next Soft Control 1 RAM address (AC 3.1).



## SOFT CONTROL 2

This 32-bit 64-word RAM provides data control for the A and C streams. RAM is started by Soft Control 1 bit 12, stepped by Soft Control 1 bit 6, or run by the response from LM.

<u>Bit</u>	<u>Description</u>
0	Clear A/C1 Buffer Register (AC 3.9).
1	Enable C Stream End (AC 3.7).
2	Enable Byte Shift Data into A/C1 Buffer Register (AC 3.9).
3-6	Control Formation of Mark Bits (AC 3.17).
7	Enable load AC Micrand Register during a load byte operation which does not cross a word boundary (AC 3.0).
8	Enable load AC Micrand Register during a load byte operation which crosses a word boundary (AC 3.0).
9	Not used.
10	Load A/C1 Buffer Register (AC 3.5).
11	Enables bit and byte shifters to perform store bit operation (AC 3.8). Complement Byte Shift Count (AC 3.14).
12	Clear Load Byte Operation FF (AC 3.0). Enable Clear A1 Buffer Full FF (AC 3.5).
13	Not used.
14	Enable set End A Stream 1 FF (AC 3.7).
15	Enable Soft Control 2 address change when Stream A Response is received (AC 3.2).
16	Not used.
17	Send AC Response to ICP and LM (AC 3.4).
18	Send AC Response to ICP and LM for test and set bit instruction (AC 3.4).
19	Enable increment Byte Shift Counter (AC 3.9).
20	Hold AC A Stream Busy (AC 3.4).
21	Forces AC Function Code to read and set lock or read and clear lock operation (LM 3.2).
22, 23	A/C Stream Assembly Mux input select code (AC 3.8).
24	Enable Final Load Bit to A1 Buffer Register (AC 3.9).
25	Load A/C Stream Assembly Register (AC 3.8).
26-31	Complement of next Soft Control 2 RAM address (AC 3.2).

### PSEUDO SOFT CONTROL 3

This hardware decoder provides minor cycle address and data control for the B stream.

<u>Bit</u>	<u>Description</u>
0	Block A/B Operand to Address Adder input B (AC 3.14).
1	Enable Address Adder to B Stream Address Counter (AC 3.15). Block B Stream Address Counter to LM Byte Address Select Mux (AC 3.11).
2	Load B Stream Disassembly Counter (AC 3.12).
3	Send Load Ak Descriptor to BDP (AC 3.10). Load B Stream Length Counter (AC 3.13). Enable feedback path to Address Adder input A (AC 3.14).
4	Enable set B Stream Memory Request FF (AC 3.11).
5	Send Latch Immediate Data to BDP (AC 3.3).
6	Enable set Clear B Stream FF (AC 3.10). Enable decrement B Stream Length Counter (AC 3.13).
7	Enable load Final B Stream Byte Number Register (AC 3.12).
8	Send AC Response to ICP and LM (AC 3.4).
9	Hold B Stream Busy (AC 3.10).
10-15	Next Pseudo Soft Control Address to be decoded (AC 3.3).

### SOFT CONTROL ADDRESS SEQUENCES

The following lists the address sequence within the Soft Control 1, 2, and 3 for each AC operation. Addresses are numbered 0 through 77 (octal).

#### Store Word

SC 1	0
SC 2	0
SC 3	0

#### Load Bytes

SC 1	4-7, 2, 0
SC 2	4-7, 0 (word boundary not crossed) 4, 45-53, 0 (word boundary crossed)
SC 3	0

#### Store Bit or Test and Set Bit

SC 1 10-13, 70-77, 74-77 (repeats 74-77 until A stream end)  
SC 2 10-13, 40-44, 0  
SC 3 0

#### Store Bytes

SC 1 14-17, 1, 0 (word boundary not crossed)  
14-17, 1, 75-77, 74-77 (word boundary crossed; repeats 74-77 until A stream end)  
SC 2 14-17, 0 (word boundary not crossed)  
14, 15, 56-60, 0 (word boundary crossed)  
SC 3 0

#### A Stream Load Bit

SC 1 20-23, 0  
SC 2 20-22, 0  
SC 3 0

#### A Stream No Operation

SC 1 24-26, 0  
SC 2 0  
SC 3 0

#### BDP Load/Store Right-to-Left

SC 1 30-33, 74-77 (repeats 74-77 until A stream end)  
SC 2 30-32 (load; repeats 30-32 until A stream end)  
70 (store first word; word boundary crossed)  
71 (store middle words; word boundary crossed)  
73, 0 (store last word; word boundary crossed)  
72, 0 (store; word boundary not crossed)  
SC 3 30-33, 74-77 (repeats 74-77 until B stream end)

BDP Load/Store Left-to-Right

SC 1 34-37, 74-77 (repeats 74-77 until A stream end)

SC 2 34-36 (load; repeats 34-36 until A stream end)  
74 (store first word; word boundary crossed)  
75 (store middle words; word boundary crossed)  
77, 0 (store last word; word boundary crossed)  
76, 0 (store; word boundary not crossed)

SC 3 34-37, 74-77 (repeats 74-77 until B stream end)

BDP Store Prevalidate

SC 1 40-43, 0

SC 2 0

SC 3 0

A Stream Load/Store Multiple Prevalidate

SC 1 50-57, 0

SC 2 0

SC 3 0

A Stream BDP Load Binary Data

SC 1 64-67, 3, 0

SC 2 64-67, 0

SC 3 0

B Stream Load Ak Descriptor

SC 1 0

SC 2 0

SC 3 20-22, 0 (right-to-left)  
24-26, 0 (left-to-right)

#### B Stream Initialize

SC 1    0  
SC 2    0  
SC 3    64-70

#### B Stream Load BDP Scale Count or Immediate Byte

SC 1    0  
SC 2    0  
SC 3    50-52, 0

#### ADDRESS FORMATION

The Address Formation network (AC 1.0) forms the byte number portion of the system virtual address (SVA) sent to LM. The A/B Operand input to the Address Formation network contains the operand selected by the RDSa (during phase 1) or RDSb (during phase 2) fields in OPI. For BDP instructions the C Operand is an A or B stream length field. AC Address Offset, typically an immediate operand from the instruction, is a third address component. In the C170 mode, the Address Formation network performs 18-bit and 21-bit one's complement addition or subtraction. Operands are 18-bit quantities and CM reference address in CM is 21 bits. Also in C170 mode, Address Formation network performs 32-bit two's complement addition for extended memory references. In the C180 mode, Address Formation network performs 32-bit two's complement addition.

The byte number for the Address Formation network merges with an Active Segment Identifier (ASID) from SM or from the A/C and B Stream ASID Registers. Together they comprise AC Address, which is an SVA.

#### LOAD STORE MULTIPLE LENGTH ADDER

This adder computes the total number of words to be transferred during a load/store multiple registers instruction. The Address Adder then adds this number to the first address of the block to be transferred. This last word address is used in the page prevalidation process. C Operand Bits 48 through 63 contain the following Xk Register information.

<u>Bits</u>	<u>Description</u>
48-51	First A register to be transferred
52-55	First X register to be transferred
56-59	Last A register to be transferred
60-63	Last X register to be transferred

The adder determines the total number of registers to be transferred by:

- Subtracting First A Register number From last A Register number.
- Subtracting First X Register number From last X Register number.
- Adding A Register total result plus X Register total result plus one if A and X Registers are to be transferred, passing A Register total result if only A Registers are to be transferred, or passing X Register total result if only X Registers are to be transferred.

The adder sends this transfer length to the Address Adder which adds it to the first word address of the memory block.

#### ADDRESS SELECT MUX

This mux selects one of three inputs to the Address Adder.

- Load/Store Multiple Length Adder in AC.
- C Operand from OPI (byte or BDP length field).
- AC Address Offset from OPI (Q, D, or BDP offset field).

#### ADDRESS ADDER

This device usually operates as an 18/21-bit (C170 mode) or 32-bit (C180 mode) adder. There are three possible external inputs.

- Address Select Mux in AC.
- AC Address Offset from OPI (Q, D, or BDP offset field).
- A/B Operand from OPI (Xj, Xk, Aj, or Ak).

The result may be fed back to one of the inputs and added to whatever is selected to the other input. This allows three quantities to be added to form a result in two minor cycles.

#### A/C, B STREAM ADDRESS COUNTERS

These counters hold the current word address for the A/C and B streams. Upon LM acceptance of an address, the Address Counter increments (left-to-right addressing) or decrements (right-to-left addressing) to update the word address.

#### A/C, B STREAM ACTIVE SEGMENT IDENTIFIER REGISTERS

These registers hold the active segment identifier (ASID) for the A/C and B stream SVA. They load from SM when a segment is initially accessed, and hold the ASID during the time it takes to read (A,B) or store (C) the entire data stream.

#### LM SYSTEM VIRTUAL ADDRESS AND BYTE ADDRESS SELECT MUX

This mux selects one of three byte number addresses, links the selected address with the ASID from SM, and sends this AC Address to LM. The three addresses are selected from:

- Address Adder.
- A/C Stream Address Counter.
- B Stream Address Counter.

#### A/C, B STREAM LENGTH COUNTERS

These counters indicate the number of words remaining to be read (A, B stream) or written (C stream) in CM during a BDP instruction. The counters decrement each time LM accepts a word address. Transfer termination is enabled when the count reaches minus one (FF<sub>16</sub> on wrap-around).

#### C170 MODE RANGE TESTER

This tester evaluates all C170 operand addresses to ensure they are in range. If operand address plus CM reference address (RAC) exceeds CM field length (FLC) plus RAC or the Address Adder overflows, the tester sends AC Address Out of Range to ICP.

#### RECOVERY ADDRESS MUX/REGISTER

This mux/register sends all of the A/C and B Stream Addresses to the Untranslatable Pointer (UTP) register in ICP. The UTP register latches the address if an address exception occurs.

#### WORD AND PAGE BOUNDARY CONTROL

This circuit detects when an AC sequence crosses a CM word boundary. When this occurs, a different soft control sequence may be needed. Also, along with LM, this circuit determines if a page boundary is crossed during a prevalidate operation. Prevalidated Page indicates to LM that the page is in CM and that its system page table status is current.

#### ERROR DETECTOR

This detector monitors various conditions in AC for possible errors and exceptions. Operand Address Specification Error indicates Address Adder overflow occurred or a word operation is not starting in byte 0. AC Before Point Of No Return Processor Detected Malfunction indicates micrand, soft control, or length counter parity error occurred.

#### MARK GENERATOR

This circuit determines which bytes of a word are to be written in CM. Mark bits 0 through 7 enable writing bytes 0 through 7 at a given word address.

#### ALN SHIFT COUNT REGISTER

This is a holding register for quantities used, but not able to be held, by ALN. ALN uses this register to hold:

- Shift Counts.
- Bit String Descriptor (C180 AC through AE Instructions).
- Exponent for C170 Pack Instructions.

#### A/C STREAM DISASSEMBLY/ASSEMBLY

This network (AC 1.0) provides one of the input data paths (A stream) from LM to BDP and the output data path (C stream) from BDP to LM. The A stream disassembles 64-bit words from LM into 8-bit bytes for BDP. Conversely, the C stream assembles 8-bit bytes into 64-bit words for LM. Data flows between LM and BDP at a rate of one byte per minor cycle. Load and store operations (except load word) also use the A/C Stream Disassembly/Assembly network.

#### A/C STREAM ASSEMBLY MUX/REGISTER

This mux/register assembles 8-bit bytes from C stream stage 5 in BDP into 64-bit words for LM under control of the A/C Stream Disassembly/Assembly Counter. This circuit may also select LM Read Data (for A stream or load operations) or C Operand (for store operations). If selected, these inputs pass through unchanged as Assembly Register Data. The register provides the first stage of buffering for A or C stream data.

#### LOAD/STORE SHIFT CONTROL

This control processes all bit and byte operations under control of Bit Right Shift Counter and Byte Shift Counter. BDP A and C stream data and load/store word data pass through unchanged.

#### BIT RIGHT SHIFT COUNTER

This counter functions as a register to hold the Bit Right Shift Count needed to execute load/store bit and test and set bit operations. It also holds data used by the Mark Generator for load/store byte and BDP operations. This counter decrements once for BDP operations.

#### BYTE SHIFT COUNTER

This counter determines the Byte Shift Count needed to execute load/store bit/byte and test and set bit operations.



#### A/C1 AND A1 BUFFER REGISTERS

The A/C1 Buffer Register serves as the second stage buffer for the A and C streams and a holding register for store data. Data from BDP (C stream) and store data proceed to LM as LM Write Data. Mark bits determine which bytes are written into memory.

The A1 Buffer Register serves as the third stage buffer for A stream data and a holding register for load data.

#### A STREAM DISASSEMBLY MUX

This mux disassembles 64-bit words from LM into eight 8-bit bytes for BDP (A stream) under control of the A/C Stream Disassembly/Assembly Counter.

#### A/C STREAM DISASSEMBLY/ASSEMBLY COUNTER

This counter controls A stream disassembly of 64-bit words from LM into 8-bit bytes for BDP. Conversely, it controls C stream assembly of 8-bit bytes from BDP to 64-bit words for LM. Initially, the rightmost three address bits of the first byte to be transferred load into the counter.

#### FINAL A/C STREAM BYTE NUMBER REGISTER

This register holds the rightmost three address bits of the last byte to be disassembled/assembled in the A/C stream. For left-to-right memory operations, this is the rightmost byte of the string. For right-to-left memory operations, this is the leftmost byte. This register is used for BDP operations only.

#### A/C STREAM CONTROL

A/C Stream Control provides hardware control for the A/C stream. The following paragraphs describe the control signals.

#### A/C Disassembly/Assembly Count Equals Final Byte Number

This signal indicates that the contents of the A/C Stream Disassembly/Assembly Counter equals the contents of the Final A/C Stream Byte Number Register.

#### AC A Stream Go

This signal from ICP initiates an A stream operation.

#### Pause A Stream

This signal from BDP causes the A stream to temporarily stop sending data to BDP. This allows synchronization of the A and B streams within BDP.

#### C Stream Stage 4 Active

This signal from BDP alerts AC that C stream data will soon be arriving from BDP.

#### Stream A Accept

This signal from LM indicates that an A stream address has been accepted by LM.

#### Stream A/C Response

This signal from LM indicates LM Read Data is available for the A stream or C stream data is accepted by LM.

#### AC Response

This signal to ICP and LM indicates AC has received a micrand and completed the required operation.

#### AC A/C Stream Busy

This signal to ICP indicates an A or C stream operation is in process.

#### Test Address

This signal to LM initiates a page table search to determine if a certain page resides in CM.

#### A/C Stream LM Request

This signal to LM initiates a CM reference for data associated with the A/C stream.

#### Load Aj Descriptor

This signal to BDP enables loading the A stream BDP descriptor into the Aj Length and Type Registers in BDP.

#### Pause C Stream

This signal to BDP stops C stream data whenever AC is unable to accept it.

#### A Stream Go

This signal to BDP indicates the A stream data lines to BDP are active or exhausted.

#### A Stream Last Byte

This signal to BDP indicates the last byte of the A stream is currently being sent to BDP.

#### A Stream Exhausted

This signal to BDP activates one minor cycle after the A stream Last Byte signal.

#### A/C Length Equals Minus One

This signal indicates the A/C Length Counter has decremented from zero to minus one. This enables the A/C stream end sequence.

#### B STREAM DISASSEMBLY

This network (AC 1.0) provides the second input data path from LM to BDP. The B stream disassembles 64-bit words from LM into eight 8-bit bytes for BDP. Data flows from LM to BDP at a rate of one byte per minor cycle.

#### B AND B1 STREAM BUFFER REGISTERS

These registers serve as the first and second buffers for B stream data.

#### B STREAM DISASSEMBLY MUX

This mux disassembles 64-bit words from LM into 8-bit bytes for BDP (B stream) under control of the B Stream Disassembly Counter.

#### B STREAM DISASSEMBLY COUNTER

This counter controls B stream disassembly of 64-bit words from LM into 8-bit bytes for BDP. Initially, the rightmost three address bits of the first byte to be transferred load into the counter.

#### FINAL B STREAM BYTE NUMBER REGISTER

This register holds the rightmost three address bits of the last byte to be disassembled in the B stream. For left-to-right memory operations, this is the rightmost byte of the string. For right-to-left memory operations, this is the leftmost byte. This register is used for BDP operations only.

## B STREAM CONTROL

B Stream Control provides hardware control for the B stream. The following paragraphs describe the control signals.

### B Disassembly Count Equals Final Byte Number

This signal indicates that the contents of the B Stream Disassembly Counter equals the contents of the Final B Stream Byte Number Register.

### AC B Stream Go

This signal from ICP initiates a B stream operation.

### Pause B Stream

This signal from BDP causes the B stream to temporarily stop sending data to BDP. This allows synchronization of the A and B streams within BDP.

### Stream B Accept

This signal from LM indicates a B stream address has been accepted by LM.

### Stream B Response

This signal from LM indicates LM Read Data is available for the B stream.

### AC B Stream Busy

This signal to ICP indicates a B stream operation is in process.

### B Stream LM Request

This signal to LM initiates a CM reference for data associated with the B stream.

### Load Ak Descriptor

This signal to BDP enables loading the B stream BDP descriptor into the Ak Length and Type Registers in BDP.

#### B Stream Go

This signal to BDP indicates the B stream data lines to BDP are active or exhausted.

#### B Stream Last Byte

This signal to BDP indicates the last byte of the B stream is currently being sent to BDP.

#### B Stream Exhausted

This signal to BDP activates one minor cycle after the B Stream Last Byte signal.

#### B Length Equals Minus One

This signal indicates the B Length Counter has decremented from zero to minus one. This enables the B stream end sequence.

#### BDP INSTRUCTIONS

The A/C Stream Disassembly/Assembly and B Stream Disassembly networks (AC 1.0) provide data paths and control for BDP instructions. The A and B streams carry LM Read Data from LM to BDP. For BDP load binary data operations, the A stream carries Register File data from OPI to BDP. The C stream carries LM Write Data from BDP to LM. Since the A and C streams are never busy at the same time, they share common hardware within AC.

#### A STREAM DISASSEMBLY

This network transfers 64-bit words from LM to BDP one byte at a time as follows:

1. A/C Stream Assembly Mux selects 64-bit LM Read Data or Register File word to enter Register.
2. Assembly Register Data passes through Load/Store Shift Control unchanged and becomes Byte Shift Data.
3. Byte Shift Data passes through A/C1 and A1 Buffer Registers as A Stream Data.
4. A Stream Disassembly Mux sends 8-bit A Stream Data byte to BDP at 16-nanosecond rate (maximum) under control of A/C Stream Disassembly/Assembly Counter which increments for left-to-right operations and decrements for right-to-left operations.
5. Step 4 repeats until all eight bytes have been sent to BDP or A/C Stream Length Counter decrements to last word and A/C Stream Disassembly/Assembly Count Equals Final Byte Number.
6. Steps 1 through 5 repeat until all bytes have been sent to BDP.

## B STREAM DISASSEMBLY

This network transfers 64-bit words from LM to BDP one byte at a time as follows:

1. LM Read Data (64-bit word) passes through B and B1 Stream Buffer Registers as B Stream Data.
2. B Stream Disassembly Mux sends 8-bit B Stream Data byte to BDP at 16-nanosecond rate (maximum) under control of B Stream Disassembly Counter which increments for left-to-right operations and decrements for right-to-left operations.
3. Step 2 repeats until all eight bytes have been sent to BDP or B Stream Length Counter decrements to last word and B Stream Disassembly Count Equals Final Byte Number.
4. Steps 1 through 3 repeat until all bytes (maximum 256) have been sent to BDP.

## C STREAM ASSEMBLY

This network transfers 64-bit words to LM from BDP one byte at a time as follows:

1. A/C Stream Assembly Mux selects 8-bit C Stream Stage 5 Data byte to enter Register.
2. Step 1 repeats at a 16-nanosecond rate until all eight bytes have entered A/C Stream Assembly Register.
3. Assembly Register Data passes through Load/Store Shift Control unchanged and becomes Byte Shift Data.
4. Byte Shift Data passes through A/C1 Buffer Register enroute to LM as a 64-bit LM Write Data word. A Mark bit accompanies each byte to be written into CM. If less than eight bytes are being sent to LM, only the applicable Mark bits are sent.
5. Steps 1 through 4 repeat until all bytes (maximum 256) have been assembled and sent to LM. Termination occurs when A/C Stream Length Counter decrements to last word and A/C Stream Disassembly/Assembly Counter equals final byte number.

## LOAD/STORE INSTRUCTIONS

These instructions also use the A/C Stream Disassembly/Assembly hardware (AC 1.0). They involve transferring a single bit, a byte string, a word, or multiple words between register file locations and CM locations.

### LOAD

The A/C stream transfers data from CM to an A or X Register in OPI as follows.

1. A/C Stream Assembly Mux selects LM Read Data from OPI to enter Register.
2. Load/Store Shift Control processes Assembly Register Data differently for load byte and load bit instructions.
  - a. For load word instructions, data transfers from LM directly to Register File in OPI. AC does not transfer load word data.

- b. For load byte instructions, Load/Store Shift Control right shifts (end-around) bytes according to Byte Shift Count and outputs Byte Shift Data.
  - c. For load bit instructions, Load/Store Shift Control locates Final Load Bit in Assembly Register Data using Byte Shift Counter and Bit Right Shift Counter. Address Adder Bits 29 through 31 contain byte number. C Operand (X0) Bits 61 through 63 contain bit position within byte.
- 3. A/C1 Buffer Register selects Byte Shift Data (load byte) for entry.
- 4. Load Data passes through A1 Buffer Register enroute to selected A or X Register in OPI. Final Load Bit enters directly into A1 Buffer Register without passing through A/C1 Buffer Register. Final Load Bit enters bit position 63; zeros enter bit positions 0 through 62.

#### STORE

The A/C stream transfers data from an A or X Register in OPI to CM as follows.

- 1. A/C Stream Assembly Mux selects C Operand (A or X Register) from OPI to enter Register.
- 2. Load/Store Shift Control processes Assembly Register Data differently for store word, store byte, and store bit instructions.
  - a. For store word instructions, Load/Store Shift Control passes data through unchanged as Byte Shift Data.
  - b. For store byte instructions, Load/Store Shift Control right shifts (end around) bytes according to Byte Shift Count and outputs Byte Shift Data.
  - c. For store bit instructions, Load/Store Shift Control tests bit 63 in Assembly Register Data. If set, Load/Store Shift Control places a one in Byte Shift Data. If clear, it places a zero. Placement is determined using Byte Shift Counter and Bit Right Shift Counter. Address Adder Bits 29 through 31 contain byte number. C Operand (X0) Bits 61 through 63 contain bit position within byte.
- 3. Marked Byte Shift Data passes through A/C1 Buffer Register enroute to LM as LM Write Data.

#### TEST AND SET BIT INSTRUCTION

The A/C stream processes this two-stage instruction as a store bit followed by a load bit operation, except that store bit is unconditionally set in CM.

#### AC SEQUENCES

The following paragraphs describe the sequence of events for various AC operations. These descriptions are keyed to level 3 diagrams and other block and timing diagrams. Following is a list of the sequences described.

- BDP Load Right-to-Left Operation.
- BDP Store Right-to-Left Operation.

- BDP Load Binary Data Operation.
- Store Word Instruction.
- Load Bit Instruction.
- Store Bit Instruction.
- Test and Set Bit Instruction.
- Store Bytes Instruction.

#### BDP LOAD RIGHT-TO-LEFT OPERATION

This operation transfers up to 256 bytes from consecutive addresses in CM to BDP through the A or B stream in AC. The byte at the highest (rightmost) address transfers first, followed by the remaining bytes consecutively in descending order. The Address Adder in AC forms the rightmost CM byte address by adding:

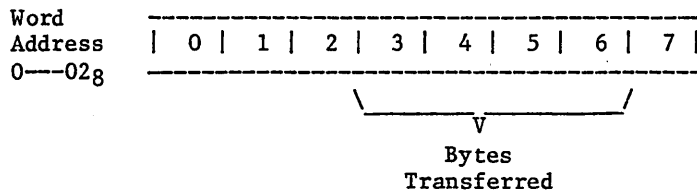
- Byte number field in Aj Register.
- Offset field in BDP descriptor.
- Length field in BDP descriptor minus one.

Figure 4-16 shows timing and the following steps describe the sequence of events for a typical 4-byte transfer through the A stream. Parameters for this example are:

$Aj + \text{Offset} = 0\text{---}023_8$  (leftmost byte address)

Length = 4

$\text{Leftmost} + (\text{Length}-1) = 0\text{---}026_8$  (rightmost byte address)





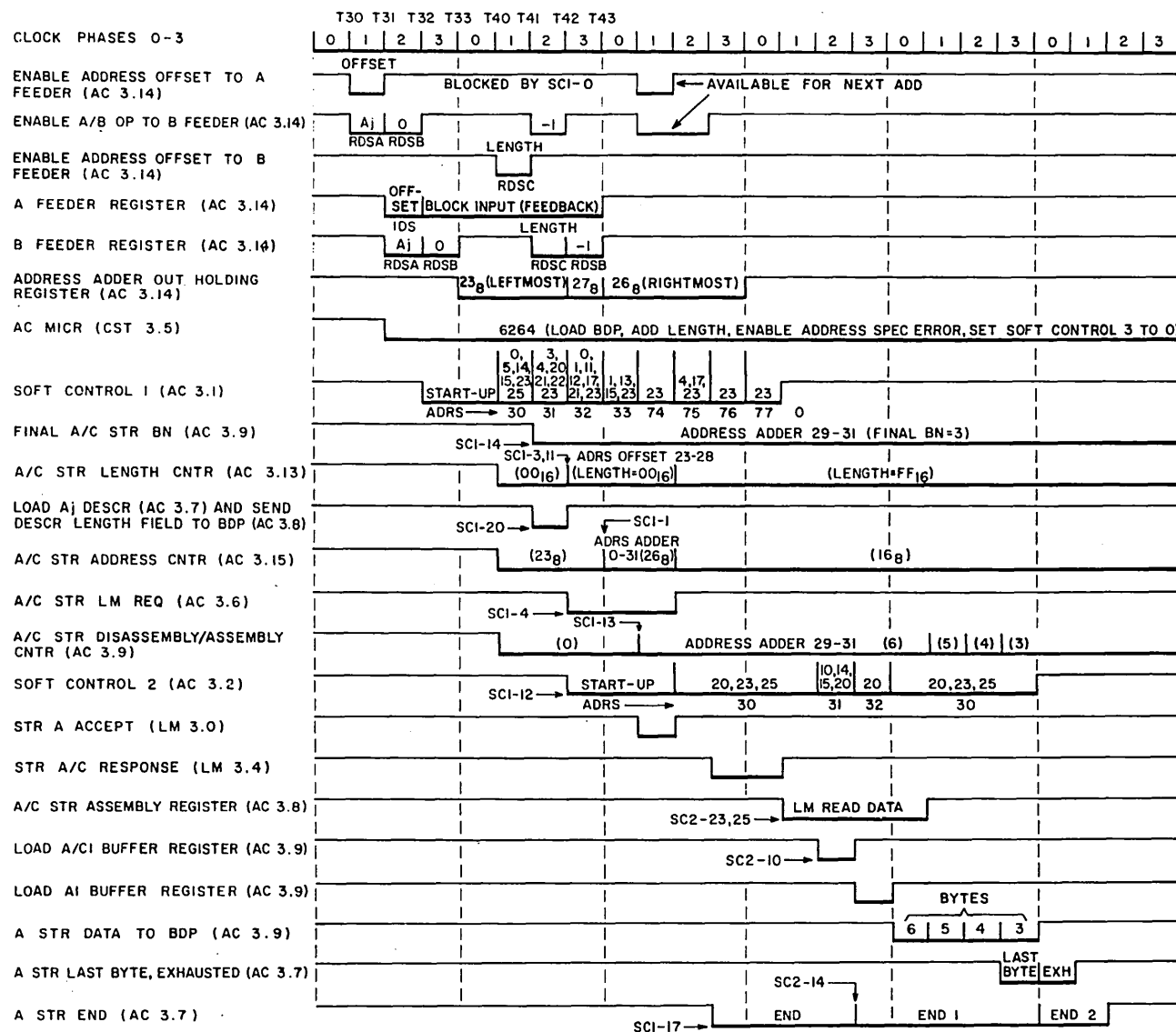


Figure 4-16. BDP Load Right-to-Left Operation

1. Functional Unit Micrand Register (CST 3.5) sends bytes 0 and 1 of Functional Unit Micrand (6264) to AC Micrand Register (AC 3.0).
2. AC Address Offset Mux (OPI 3.8) sends BDP descriptor offset field (AC Address Offset selected by immediate data select (IDS) field in general micrand) to input A of Address Adder (AC 3.14) through Address Select Mux (AC 3.13). Operand Data Mux (OPI 3.10) sends byte number in Aj Register [A/B Operand selected by Read Data Select (RDS)a field in general micrand] to input B of Address Adder.
3. Address Adder forms sum of Aj byte number and offset (leftmost byte address 0---023<sub>8</sub>).
4. Soft Control 1 (AC 3.1) activates.
5. Operand Data Mux (OPI 3.10) sends BDP descriptor length field (A/B Operand selected by RDSc field in general micrand) to input B of Address Adder. Output of Address Adder (0---023<sub>8</sub>) feeds back to input A.
6. Leftmost byte number (3) enters Final A/C Stream Byte Number Register (AC 3.9).
7. Address Adder forms sum of 0---023<sub>8</sub> and length field 0---04<sub>8</sub> (0---027<sub>8</sub>).
8. C Operand Register (OPI 3.11) sends Descriptor Length Field (selected by RDSc field in general micrand) to Aj Length Register (BDP 3.2) through Fanout (AC 3.8).
9. Address Adder subtracts 0---01<sub>8</sub> from 0---027<sub>8</sub> to form rightmost byte address (0---026<sub>8</sub>).
10. A/C Stream Length Counter (AC 3.13) sets to 0 which means one word is required from LM.
11. A/C Stream Memory Request Control (AC 3.6) sends A/C Stream LM Request to LM 3.0.
12. Soft Control 2 (AC 3.2) activates.
13. Rightmost byte address 0---026<sub>8</sub> enters A/C Stream Address Counter (AC 3.15).
14. LM System Virtual Address Select Mux (AC 3.15) sends AC Address 0---026<sub>8</sub> to Address Mux (LM 3.5).
15. Accept Control (LM 3.0) sends Stream A Accept to AC 3.0.
16. Byte count (6) enters A/C Stream Disassembly/Assembly Counter (AC 3.9) (first byte to be sent to BDP).
17. A/C Stream Length Counter (AC 3.13) decrements to minus one. This means no more words are required from LM.
18. End A Stream FF (AC 3.7) sets to initiate end sequence.
19. A/C Stream Address Counter (AC 3.15) decrements to 0---016<sub>8</sub>. This count is not used because no more words are required in this example.
20. Response Translator (LM 3.4) sends Stream A/C Response to AC 3.0.
21. LM Read Data Mux (LM 3.16) sends LM Read Data to A/C Stream Assembly Mux/Register (AC 3.8) through Pass On (OPI 3.11).

22. LM Read Data passes unchanged through Store Bit/All Other Operations Select Mux (AC 3.8) and End Around Right Byte Shifter (AC 3.8) and enters A/C1 Buffer Register (AC 3.9) as Byte Shift Data.
23. LM Read Data enters A1 Buffer Register (AC 3.9).
24. A Stream Disassembly Mux (AC 3.9) sends A Stream Data byte 6 to A Stream Data Mux (BDP 3.5) under control of A/C Stream Disassembly/Assembly Counter (AC 3.9).
25. Step 24 repeats for bytes 5, 4, and 3. Disassembly count decrements for each byte.
26. Equal Test (AC 3.9) determines that A/C Stream Disassembly/Assembly Count Equals Final Byte Number.
27. A Stream BDP Control Interface Register (AC 3.7) sends A Stream Last Byte to A Stream Status Mux (BDP 3.5).
28. A Stream BDP Control Interface Register (AC 3.7) sends A Stream Exhausted to A Stream Status Mux (BDP 3.5).

#### BDP STORE RIGHT-TO-LEFT OPERATION

This operation transfers up to 256 bytes from BDP to consecutive addresses in CM through the C stream in AC. The byte at the highest (rightmost) address transfers first, followed by the remaining bytes in descending order. The Address Adder in AC forms the rightmost CM byte address by adding:

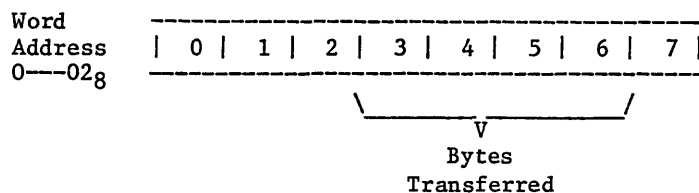
- Byte number field in Aj Register.
- Offset field in BDP descriptor.
- Length field in BDP descriptor minus one.

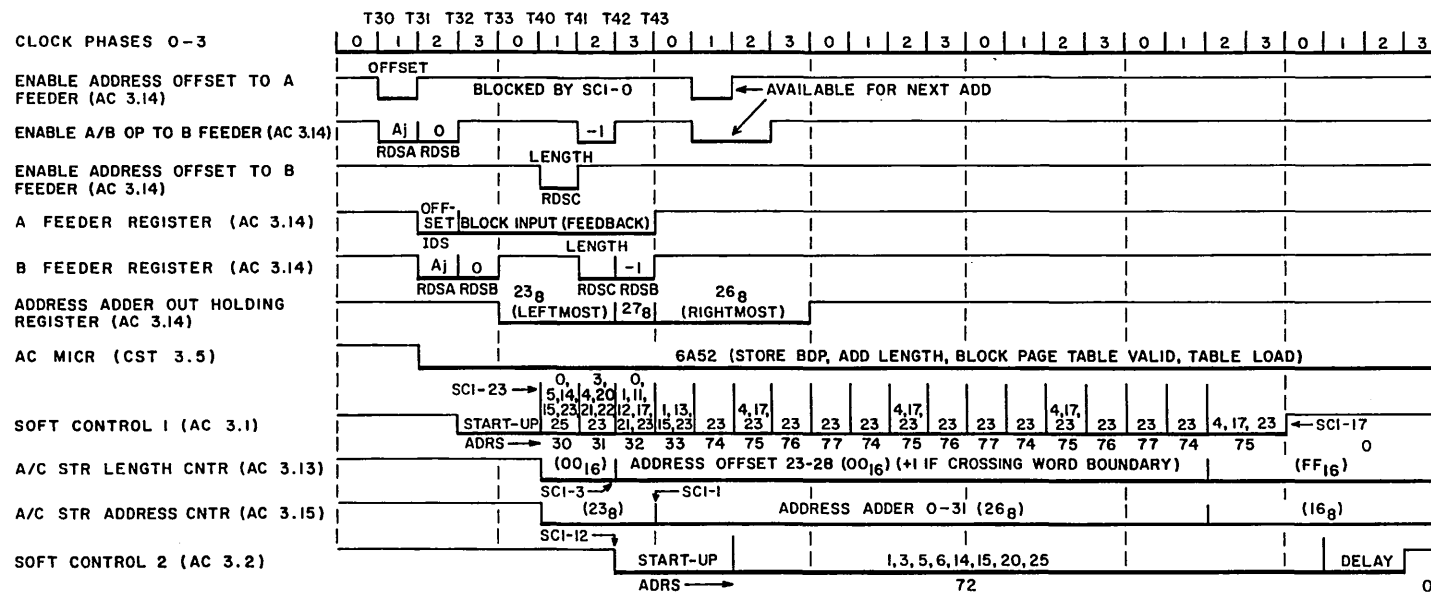
Figure 4-17 shows timing and the following steps describe the sequence of events for a typical 4-byte transfer through the C stream. Parameters for this example are:

$Aj + \text{Offset} = 0\text{---}023_8$  (leftmost byte address)

Length = 4

$\text{Leftmost} + (\text{Length}-1) = 0\text{---}026_8$  (rightmost byte address)





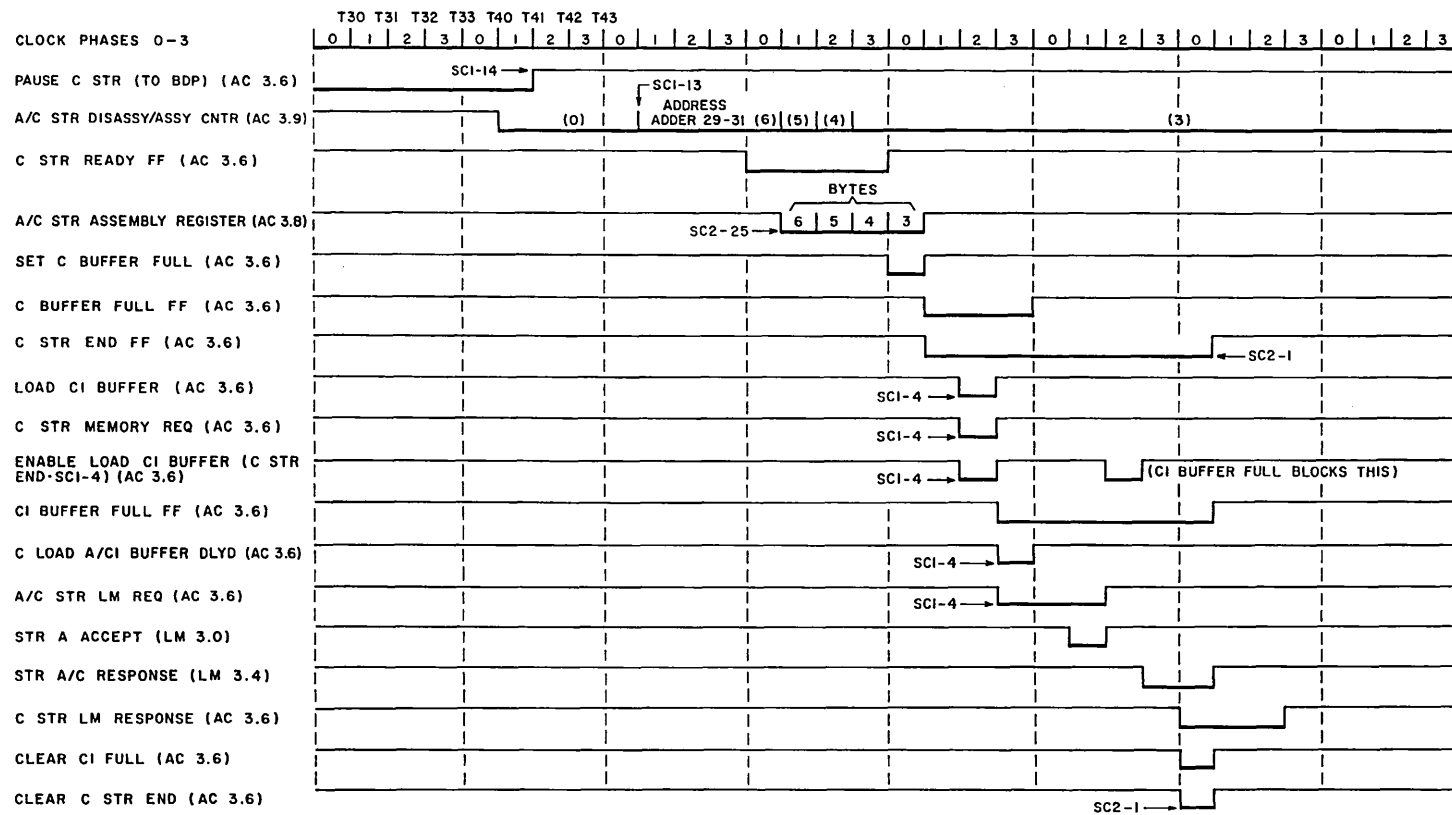


Figure 4-17. BDP Store Right-to-Left Operation (Sheet 2 of 2)

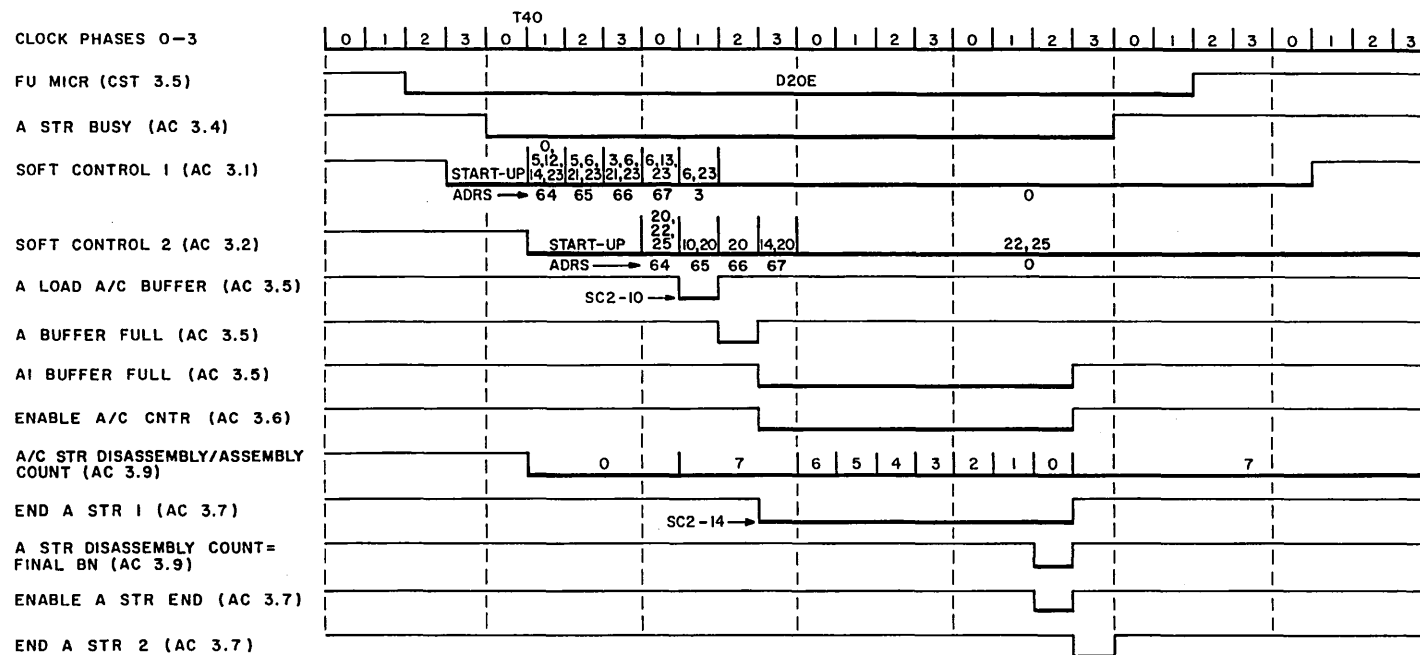
1. Functional Unit Micrand Register (CST 3.5) sends bytes 0 and 1 of Functional Unit Micrand (6A52) to AC Micrand Register (AC 3.0).
2. AC Address Offset Mux (OPI 3.8) sends BDP descriptor offset field (AC Address Offset selected by IDS field in general micrand) to input A of Address Adder (AC 3.14) through Address Select Mux (AC 3.13). Operand Data Mux (OPI 3.10) sends byte number in Aj Register (A/B Operand selected by RDSa field in general micrand) to input B of Address Adder.
3. Address Adder forms sum of Aj byte number and offset (leftmost byte address 0---023<sub>8</sub>).
4. Soft Control 1 (AC 3.1) activates.
5. Operand Data Mux (OPI 3.10) sends BDP descriptor length field (A/B Operand selected by RDSc field in general micrand) to input B of Address Adder. Output of Address Adder (0---23<sub>8</sub>) feeds back to input A.
6. Address Adder forms sum of 0---023<sub>8</sub> and length field 0---04<sub>8</sub> (0---027<sub>8</sub>).
7. C Stream Pause Control (AC 3.6) drops Pause C Stream which enables BDP to start sending data to AC.
8. Address Adder subtracts 0---01<sub>8</sub> from 0---027<sub>8</sub> to form rightmost byte address (0---026<sub>8</sub>).
9. Response Control (AC 3.4) sends AC Response to ICP 3.6.
10. A/C Stream Length Counter (AC 3.13) sets to 0 which means one word will be sent to LM.
11. Rightmost byte address 0---026<sub>8</sub> enters A/C Stream Address Counter (AC 3.15).
12. Byte count 6 enters A/C Stream Disassembly/Assembly Counter (AC 3.9) (first byte to be sent to LM).
13. Soft Control 2 (AC 3.2) activates.
14. C Stream Ready FF (AC 3.6) sets to enable assembly of bytes from BDP.
15. C Stream Output Mux (BDP 3.22) sends C Stream Stage 5 byte 6 to A/C Stream Assembly Mux (AC 3.8).
16. Byte 6 enters A/C Stream Assembly Register (AC 3.8) under control of A/C Stream Disassembly/Assembly Counter (AC 3.9).
17. Steps 15 and 16 repeat for bytes 5, 4, and 3. Assembly count decrements for each byte.
18. C Stream End FF (AC 3.6) sets.
19. Assembled bytes enter A/C1 Buffer Register (AC 3.9).
20. A/C Stream Memory Request Control (AC 3.6) sends A/C Stream LM Request to LM 3.0.

21. Accept Control (LM 3.0) sends Stream A Accept to AC 3.0.
22. Response Translator (LM 3.4) sends Stream A/C Response to AC 3.0.
23. LM Write Data from A/C1 Buffer Register (AC 3.9) enters LM Write Data Register (LM 3.16).

#### BDP LOAD BINARY DATA OPERATION

This operation transfers up to eight bytes of a word in the Register File (OPI) to BDP through the A stream in AC. The byte at the highest (rightmost) address transfers first, followed by the remaining bytes consecutively in descending order. Figure 4-18 shows timing and the following steps describe the sequence of events for transferring the last word during a BDP decimal product or quotient instruction.

1. C Operand Register (OPI 3.11) sends literal 7 (C Operand selected by RDSb field in general micrand) to input B of Address Adder (AC 3.14).
2. Address Adder forms sum of 0---0<sub>8</sub> and 0---7<sub>8</sub> to form rightmost byte address (0---7<sub>8</sub>).
3. Functional Unit Micrand Register (CST 3.5) sends bytes 0 and 1 of Functional Unit Micrand (D20E) to AC Micrand Register (AC 3.0).



**Figure 4-18. BDP Load Binary Data Operation**



4. Final A/C Stream Byte Number Register (AC 3.9) resets to 0.
5. Soft Control 1 (AC 3.1) activates.
6. Soft Control 2 (AC 3.2) activates.
7. A/C Stream Disassembly/Assembly Counter (AC 3.9) indicates 7 (first byte to be sent to BDP).
8. C Operand Register (OPI 3.11) sends Register File binary data word (C Operand selected by RDS<sub>c</sub> field in general micrand) to A/C Stream Assembly Mux/Register (AC 3.8).
9. C Operand passes unchanged through Store Bit/All Other Operations Select Mux (AC 3.8) and End Around Right Byte Shifter (AC 3.8) and enters A/C1 Buffer Register (AC 3.9) as Byte Shift Data.
10. C Operand enters A1 Buffer Register (AC 3.9).
11. End A Stream FF (AC 3.7) sets to initiate end sequence.
12. A Stream Disassembly Mux (AC 3.9) sends A Stream Data byte 7 to A Stream Data Mux (BDP 3.5) under control of A/C Stream Disassembly/Assembly Counter (AC 3.9).
13. Step 12 repeats for bytes 6, 5, 4, 3, 2, 1, and 0. Disassembly count decrements for each byte.
14. Equal Test (AC 3.9) determines that A/C Stream Disassembly/Assembly Count Equals Final Byte Number.
15. A Stream BDP Control Interface Register (AC 3.7) sends A Stream Last Byte to A Stream Status Mux (BDP 3.5).
16. A Stream BDP Control Interface Register (AC 3.7) sends A Stream Exhausted to A Stream Status Mux (BDP 3.5).

#### STORE WORD INSTRUCTION

The store word (A3jkID) instruction transfers one word from the Xk Register to a location in CM through the A/C stream in AC. The CM word address is formed by adding:

- Byte number field in Aj Register.
- D field in instruction times 8.
- Lower (rightmost) 32 bits in Xi Register times 8.

Figure 4-19 is a simplified block diagram showing the hardware needed to form the address and transfer the word to CM. The diagram includes instruction timing information which relates to the following sequence of events.

1. (T30) AC Address Offset Mux (OPI 3.8) sends D times 8 (selected by IDS field in general micrand) to input A of Address Adder (AC 3.14). Operand Data Mux (OPI 3.9) sends byte number in Aj Register (selected by RDSa field in general micrand) to input B of Address Adder and segment number to SM.
2. (T31) Operand Data Mux (OPI 3.9) sends Xi times 8 (selected by RDSb field in general micrand) to input B of Address Adder (AC 3.14). Address Adder feeds back result (D times 8 plus Aj byte number) to input A. Address Adder forms sum of D times 8, Aj byte number, and Xi times 8.
3. (T32) Functional Unit Go Control (ICP 3.5) generates AC A Stream Go Request to start Word Operation Timing Chain (AC 3.4), Go SM to enable access violation reporting in SM, and Instruction Issue Request Enable to initiate LM. Operand Data Mux (OPI 3.9) sends Xk Register data to C Operand Register (OPI 3.11).
4. (T33) LM Address Selector (AC 3.15) sends AC Address from Address Adder (Byte Number) and SM (Active Segment Identifier) to LM.
5. (T40) Xk Register data loads into C Operand Register (OPI 3.11).
6. (T41) Xk Register data loads into A/C Stream Assembly/Mux Register (AC 3.8), enabled by SC 2 bits 22 and 25 (address 0).
7. (T43) Word Operation Timing Chain (AC 3.4) sends Enable Load A/C1 Buffer to A/C1 Buffer Register (AC 3.9).
8. (T50) Xk Register data loads into A/C1 Buffer Register, which sends data to LM.

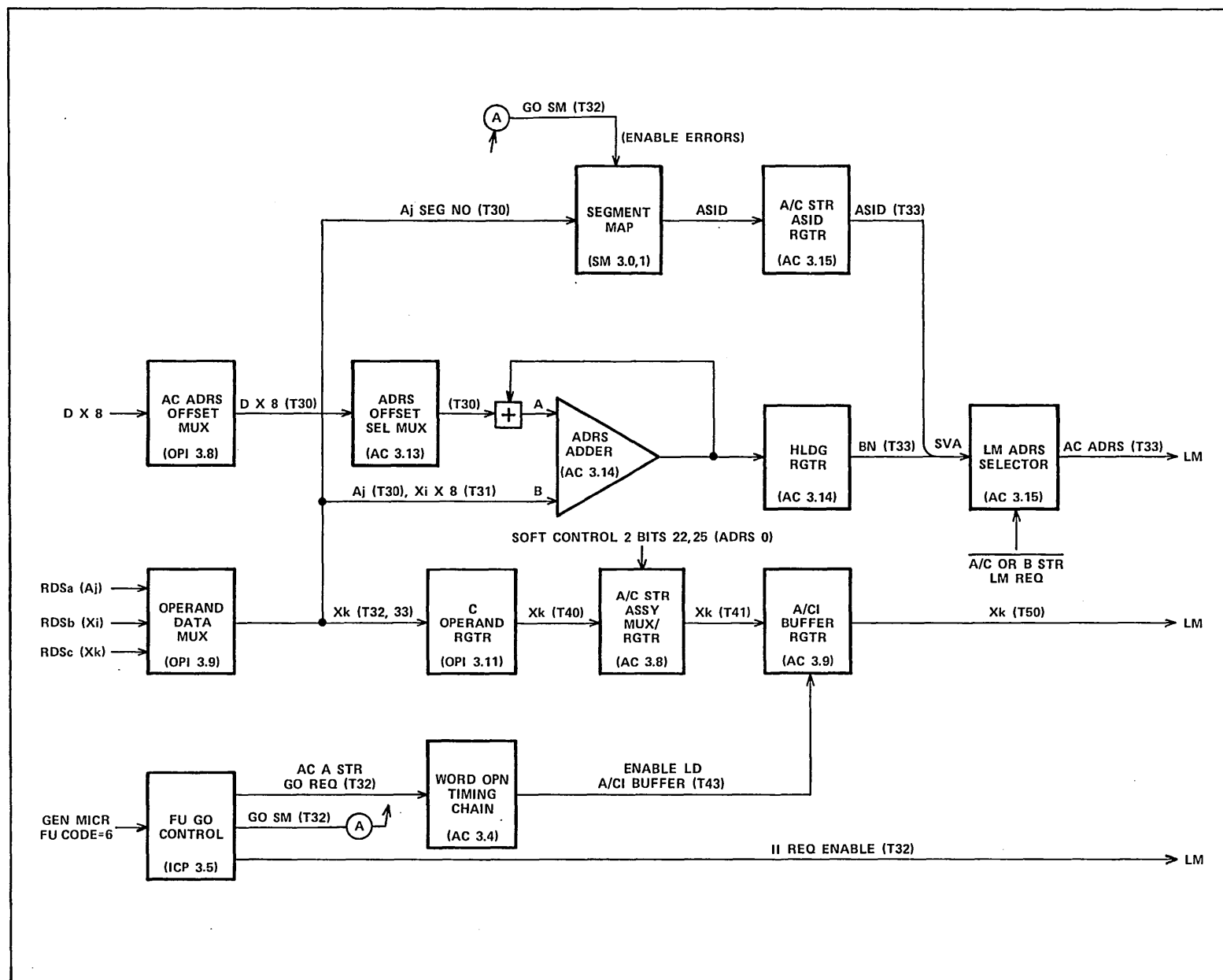


Figure 4-19. Store Word Instruction

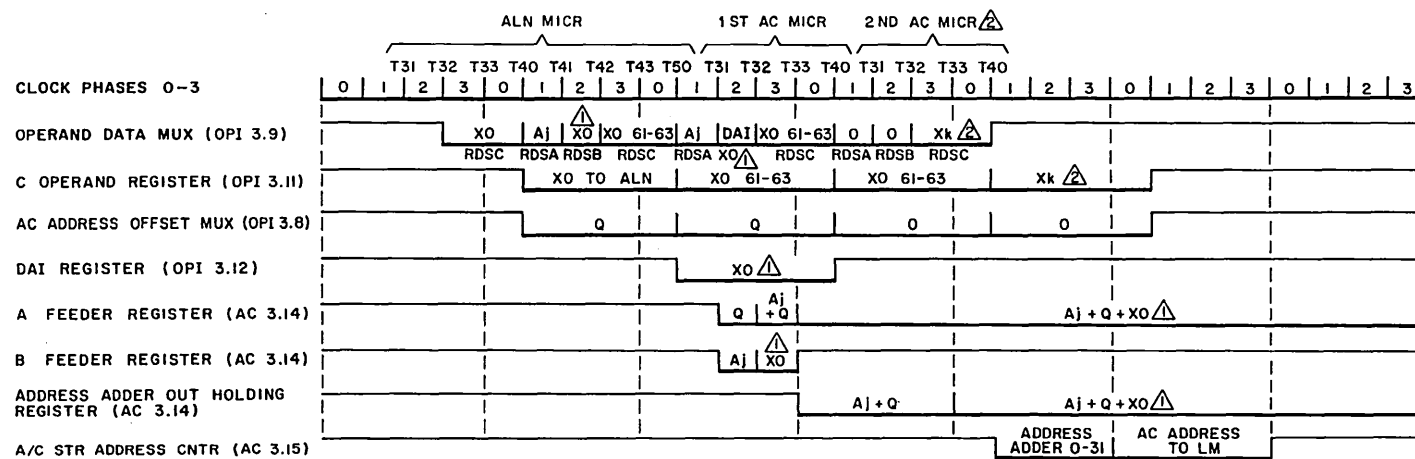
## LOAD BIT INSTRUCTION

The load bit (88jkQ) instruction sends a bit from CM to Xk Register bit position 63 through the A/C stream in AC. The Address Adder in AC forms the CM address of the byte containing the bit to be loaded by adding:

- Byte number field in Aj Register.
- Q field in instruction.
- Lower (rightmost) 32 bits in X0 Register right-shifted three bit positions, end-off, and sign-extended.

X0 Register bits 61 through 63 determine the bit position within the addressed byte to be loaded. Values of 0 through 7 select the corresponding bit positions within the byte. Figure 4-20 shows timing for address formation. Figure 4-21 shows execution timing for a typical load bit instruction where byte 6 bit 4 (set) at the addressed CM location enters X8 Register bit position 63. The following steps describe the sequence of events.

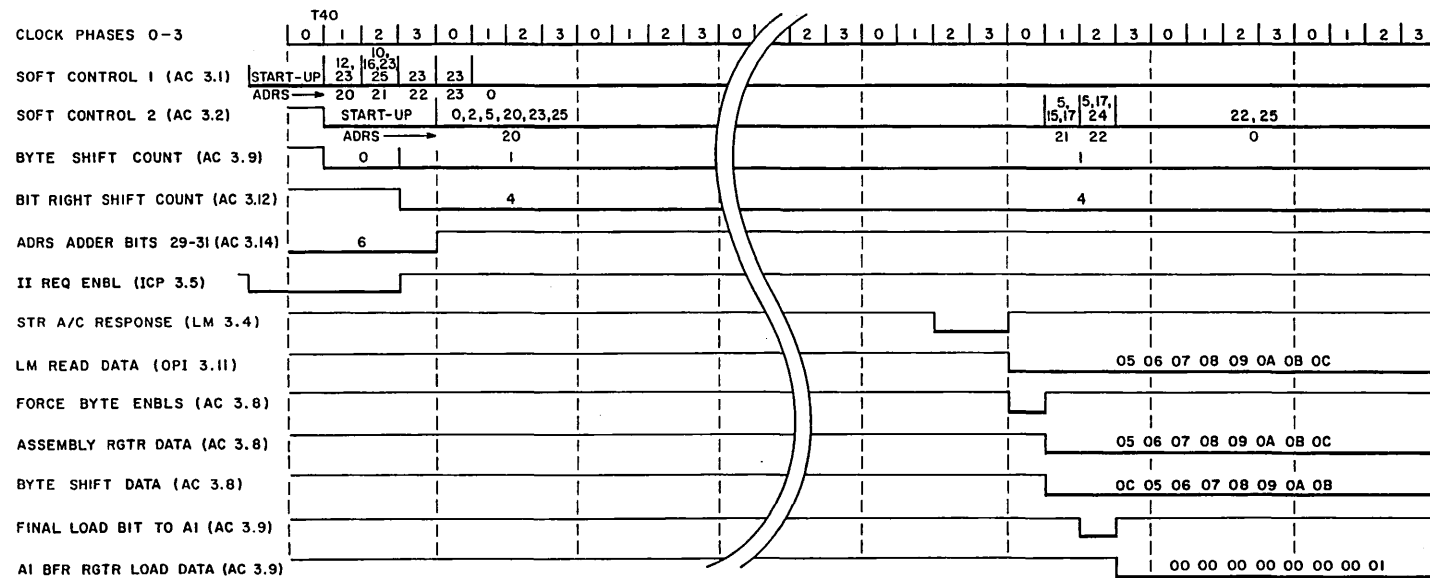
1. Functional Unit Micrand Register (CST 3.5) sends Functional Unit Micrand (0776 1E00 0000 0100) to Control/Micrand Register (ALN 3.0).
2. Operand Data Mux (OPI 3.9) sends X0 Register data from Register File (OPI 3.5) (selected by RDSb field in general micrand) to C Operand Register (OPI 3.11).
3. C Operand Register sends X0 Register data to Shift Network Mux (ALN 3.5).
4. Shifter Ranks 1 and 2 (ALN 3.6) right-shift (sign-extended) X0 Register data three bit positions.
5. ALN Output Mux Upper (ALN 3.12) and Lower (ALN 3.10) send ALN Result to Data Interchange Register (OPI 3.12).
6. Functional Unit Micrand Register (CST 3.5) sends bytes 0 and 1 of Functional Unit Micrand (4120) to AC Micrand Register (AC 3.0).
7. AC Address Offset Mux (OPI 3.8) sends instruction Q field (AC Address Offset selected by IDS field in general micrand) to input A of Address Adder (AC 3.14) through Address Select Mux (AC 3.13). Operand Data Mux (OPI 3.10) sends byte number in Aj Register (A/B Operand selected by RDSa field in general micrand) to input B of Address Adder.
8. Address Adder forms sum of Aj Register byte number and Q field.
9. LM Go Control (ICP 3.5) sends Instruction Issue Request Enable to Instruction Issue Request Control (LM 3.0).
10. Soft Control 1 (AC 3.1) activates.
11. Operand Data Mux (OPI 3.10) sends shifted X0 Register data (A/B Operand selected by RDSb field in general micrand) to input B of Address Adder (AC 3.14). Output of Address Adder feeds back to input A.
12. Address Adder forms sum of Aj Register byte number, Q field, and shifted X0 Register data.
13. Soft Control 2 (AC 3.2) activates.



## NOTES:

- $\triangle$  SHIFTED X0 SHORTSTOPPED FROM ALN.  
 $\triangle$  APPLICABLE TO STORE BIT INSTRUCTION ONLY.

Figure 4-20. Load/Store Bit Address Formation



14. Complement Control (AC 3.14) sets Byte Shift Counter (AC 3.9) to 1 which is complement of Address Adder bits 29 through 31 (byte number 6).
15. C Operand Register (OPI 3.11) sends unshifted X0 Register bits 61 through 63 (bit number 4) to Bit Right Shift Counter (AC 3.12).
16. Address Adder bits enter A/C Stream Address Counter (AC 3.15).
17. LM System Virtual Address Select Mux (AC 3.15) sends AC Address to Address Mux (LM 3.5).
18. Response Translator (LM 3.4) sends Stream A/C Response to AC 3.0.
19. LM Read Data Mux (LM 3.16) sends LM Read Data (05 06 07 08 09 0A 0B 0C) to A/C Stream Assembly Mux/Register (AC 3.8) through Pass On (OPI 3.11).
20. LM Read Data passes unchanged through Store Bit/All Other Operations Select Mux (AC 3.8) as Assembly Register data.
21. End Around Right Byte Shifter (AC 3.8) shifts byte 6 into byte position 7 (0C 05 06 07 08 09 0A 0B) according to Byte Shift Count (1).
22. Bit Right Shifter (AC 3.8) shifts byte 7 bit 4 (60) into bit position 7 (63) according to Bit Right Shift Count (4). It does this by first transposing bits (bit 56 becomes 63, 57 becomes 62, and so on) and then right-shifting end-off.
23. Bit Right Shifter sends Final Load Bit (63) (set) to A1 Buffer Register (AC 3.9) which is loaded with zeros in bit positions 0 through 62.
24. A1 Buffer Register sends Load Data (0---01) to Data Interchange Register (OPI 3.12).
25. Data Interchange Register sends X8 Register data (Load Data selected by RDSa field in general micrand) to Register File (OPI 3.5).

#### STORE BIT INSTRUCTION

The store bit (89jkQ) instruction sends Xk Register bit 63 to CM through the A/C Stream in AC. The Address Adder in AC forms the CM address of the byte containing the bit to be stored by adding:

- Byte number field in Aj Register.
- Q field in instruction.
- Lower (rightmost) 32 bits in X0 Register right-shifted three bit positions, end-off, and sign-extended.

X0 Register bits 61 through 63 determine the bit position within the addressed byte to be stored. Values of 0 through 7 select the corresponding bit positions within the byte. Figure 4-20 shows timing for address formation. Figure 4-22 shows execution timing for a typical store bit instruction where X5 Register bit 63 (clear) enters byte 3 bit 2 at the addressed CM location. The following steps describe the sequence of events.

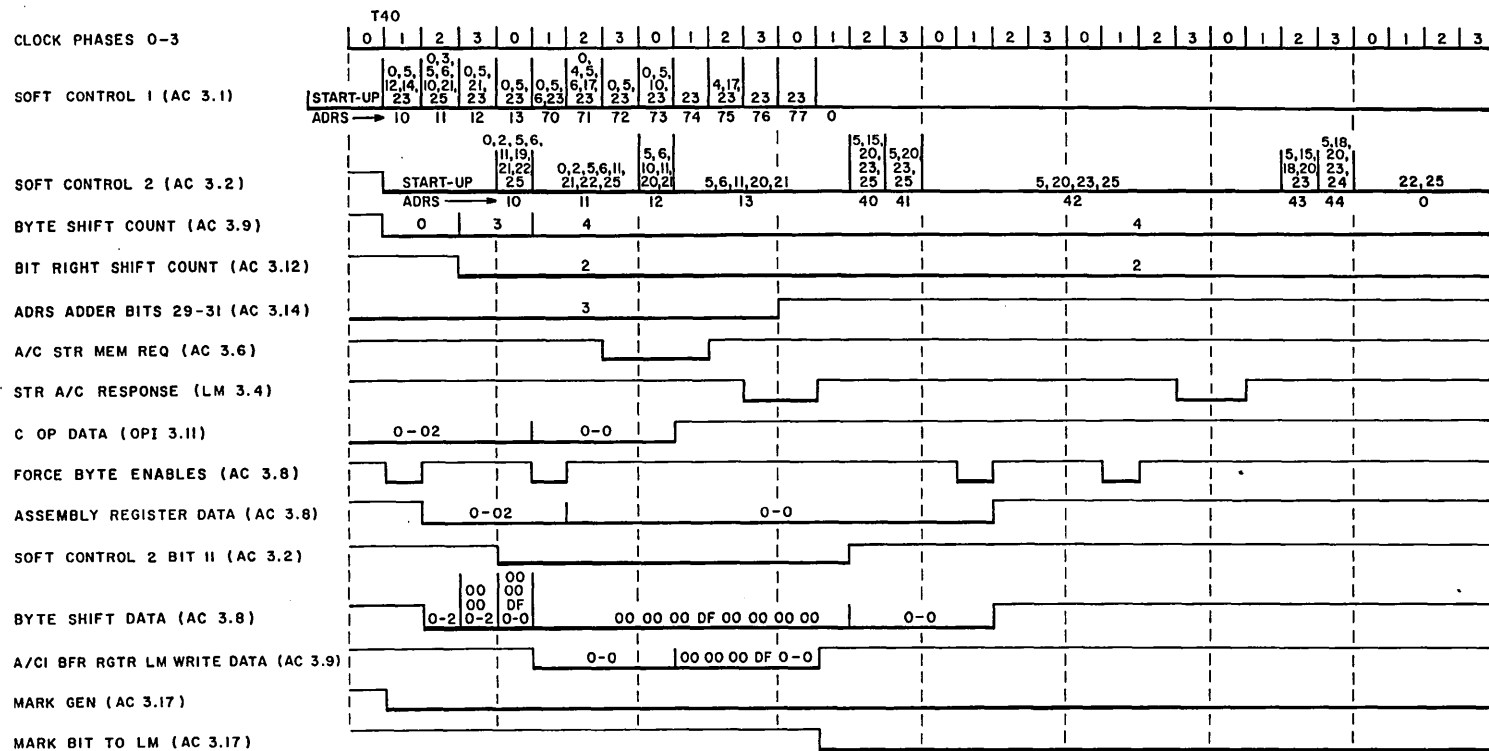


Figure 4-22. Store Bit Instruction



1. Functional Unit Micrand Register (CST 3.5) sends Functional Unit Micrand (0776 1E00 0000 0100) to Control/Micrand Register (ALN 3.0).
2. Operand Data Mux (OPI 3.9) sends X0 Register data from Register File (OPI 3.5) (selected by RDS<sub>c</sub> field in general micrand) to C Operand Register (OPI 3.11).
3. C Operand Register sends X0 Register data to Shift Network Mux (ALN 3.5).
4. Shifter Ranks 1 and 2 (ALN 3.6) right-shift (sign-extended) X0 Register data three bit positions.
5. ALN Output Mux Upper (ALN 3.12) and Lower (ALN 3.10) send ALN Result to Data Interchange Register (OPI 3.12).
6. Functional Unit Micrand Register (CST 3.5) sends bytes 0 and 1 of Functional Unit Micrand (2920) to AC Micrand Register (AC 3.0).
7. AC Address Offset Mux (OPI 3.8) sends instruction Q field (AC Address Offset selected by IDS field in general micrand) to input A of Address Adder (AC 3.14) through Address Select Mux (AC 3.13). Operand Data Mux (OPI 3.10) sends byte number in Aj Register (A/B Operand selected by RDS<sub>a</sub> field in general micrand) to input B of Address Adder.
8. Address Adder forms sum of Aj Register byte number and Q field.
9. Operand Data Mux (OPI 3.10) sends shifted X0 Register data (A/B Operand selected by RDS<sub>b</sub> field in general micrand) to input B of Address Adder (AC 3.14). Output of Address Adder feeds back to input A.
10. Address Adder forms sum of Aj Register byte number, Q field, and shifted X0 Register data.
11. Soft Control 1 (AC 3.1) activates.
12. Soft Control 2 (AC 3.2) activates.
13. Address Adder bits 29 through 31 (byte number 3) enter Byte Shift Counter (AC 3.9).
14. C Operand Register (OPI 3.11) sends unshifted X0 Register bits 61 through 63 (bit number 2) to Bit Right Shift Counter (AC 3.12).
15. Operand Data Mux (OPI 3.9) sends X5 Register data (0---0) from Register File (OPI 3.5) (selected by RDS<sub>c</sub> field in general micrand) to C Operand Register (OPI 3.11).
16. C Operand Register sends X5 Register data to A/C Stream Assembly/Mux Register (AC 3.8).
17. Address Adder bits enter A/C Stream Address Counter (AC 3.15).
18. Byte Shift Counter (AC 3.9) increments from 3 to 4.
19. Bit Right Shifter (AC 3.8) determines that Assembly Register bit 63 is clear. It then forms a byte with bit 2 clear and other seven bits set (DF).
20. Store Bit/All Other Operations Select Mux (AC 3.8) replaces Assembly Register byte 7 with byte formed by Bit Right Shifter.

21. End Around Right Byte Shifter (AC 3.8) places byte 7 into byte position 3 to form Byte Shift Data (00 00 00 DF 00 00 00 00).
22. A/C Stream Memory Request Control (AC 3.6) sends A/C Stream LM Request to LM 3.0.
23. Mark Bit Generator (AC 3.17) sets Mark bit 3.
24. LM System Virtual Address Select Mux (AC 3.15) sends AC Address to Address Mux (LM 3.5).
25. Byte Shift Data byte 3 enters A/C1 Buffer Register (AC 3.9) under control of Mark bit 3.
26. A/C1 Buffer Register sends LM Write Data (00 00 00 DF 00 00 00 00) to LM Write Data Register (LM 3.16).
27. Mark Bit Register (AC 3.17) sends AC Mark bit 3 to Mark Bit Generator (LM 3.7).
28. Response Translator (LM 3.4) sends Stream A/C Response to AC 3.0.
29. Response Translator sends second Stream A/C Response to AC. This is needed to allow Soft Control 2 to complete its cycle. Soft control is identical for store bit and test and set bit instructions.

#### TEST AND SET BIT INSTRUCTION

The test and set bit (14jk) instruction transfers one bit from a location in CM to Xk Register bit position 63. The instruction clears Xk Register bit positions 0 through 62 and sets the addressed bit in CM. The Address Adder in AC forms the CM address of the byte containing the bit to be transferred and set by adding:

- Byte number field in Aj Register.
- Lower (rightmost) 32 bits in XO Register right-shifted three bit positions, end-off, and sign-extended.

XO Register bits 61 through 63 determine the bit position within the addressed byte. Values of 0 through 7 select the corresponding bit positions within the byte. Figure 4-23 shows execution timing for a typical test and set bit instruction where byte 2 bit 6 (clear) at the addressed CM location enters X1 Register bit position 63. Byte 2 bit 6 unconditionally sets in CM. The following steps describe the sequence of events.

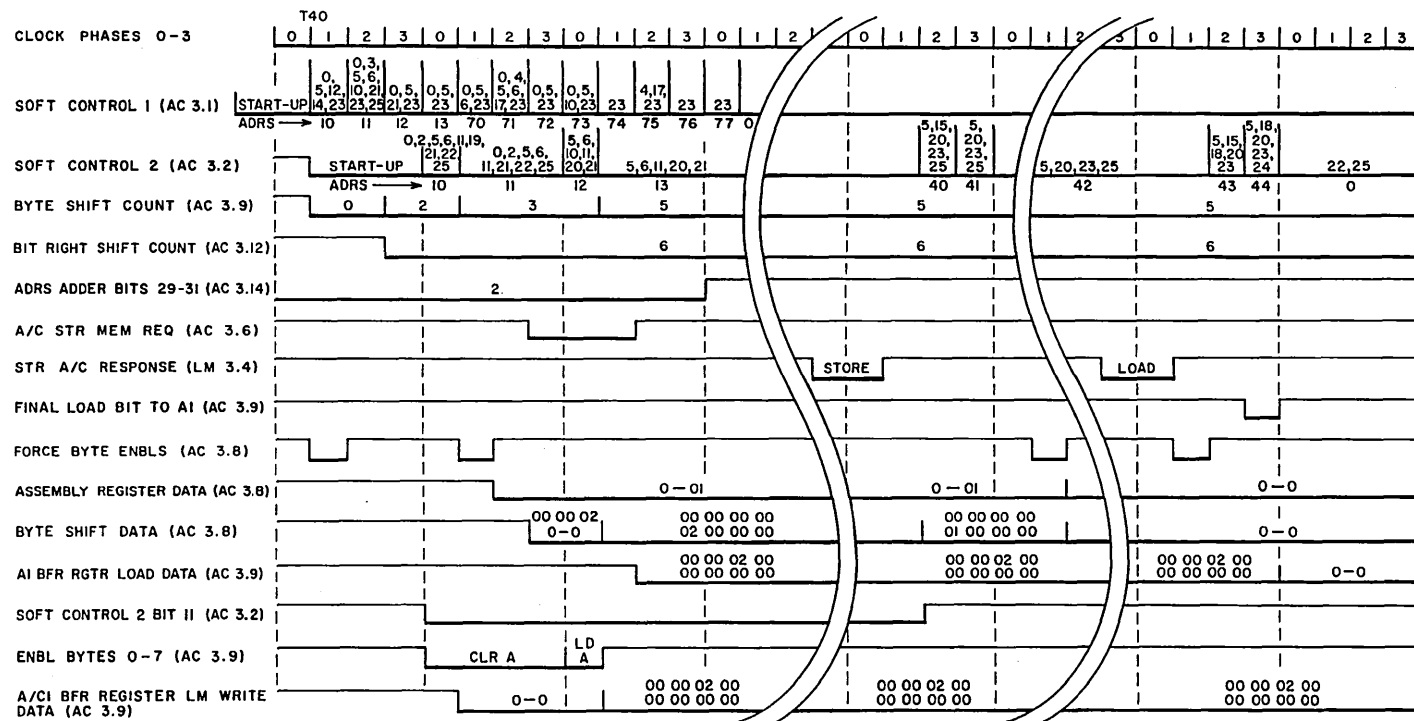


Figure 4-23. Test and Set Bit Instruction

1. Functional Unit Micrand Register (CST 3.5) sends Functional Unit Micrand (0776 1E00 0000 0100) to Control/Micrand Register (ALN 3.0).
2. Operand Data Mux (OPI 3.9) sends XO Register data from Register File (OPI 3.5) (selected by RDS<sub>c</sub> field in general micrand) to C Operand Register (OPI 3.11).
3. C Operand Register sends XO Register data to Shift Network Mux (ALN 3.5).
4. Shifter Ranks 1 and 2 (ALN 3.6) right shift (sign extended) XO Register data three bit positions.
5. ALN Output Mux Upper (ALN 3.12) and Lower (ALN 3.10) send shifted XO Register data (ALN Result selected by RDS<sub>a</sub> field in general micrand) to Register File (OPI 3.5) through Data Interchange Register (OPI 3.12).
6. Functional Unit Micrand Register (CST 3.5) sends bytes 0 and 1 of Functional Unit Micrand (2928) to AC Micrand Register (AC 3.0).
7. Operand Data Mux (OPI 3.10) sends byte number in Aj Register (A/B Operand selected by RDS<sub>a</sub> field in general micrand) to input B of Address Adder (AC 3.14).
8. Address Adder forms sum of Aj byte number and zero.
9. Operand Data Mux (OPI 3.10) sends shifted XO Register data (A/B Operand selected by RDS<sub>b</sub> field in general micrand) to input B of Address Adder. Output of Address Adder feeds back to input A.
10. Address Adder forms sum of Aj byte number and shifted XO Register data.
11. Soft Control 1 (AC 3.1) activates.
12. Soft Control 2 (AC 3.2) activates.
13. Address Adder bits 29 through 31 (byte number 2) enter Byte Shift Counter (AC 3.9).
14. C Operand Register (OPI 3.11) sends unshifted XO Register bits 61 through 63 (bit number 6) to Bit Right Shift Counter (AC 3.12).
15. Immediate Operand Mux (OPI 3.7) sends 0---01 (selected by RDS<sub>c</sub> field in general micrand) to C Operand Register (OPI 3.11) through Operand Data Mux (OPI 3.9).
16. C Operand Register sends 0---01 to A/C Stream Assembly/Mux Register (AC 3.8).
17. Address Adder bits enter A/C Stream Address Counter (AC 3.15).
18. Byte Shift Counter (AC 3.9) increments from 2 to 3.
19. Bit Right Shifter (AC 3.8) determines that Assembly Register Bit 63 is set. It then forms a byte with bit 6 set and other seven bits clear (02).
20. Store Bit/All Other Operations Select Mux (AC 3.8) replaces Assembly Register byte 7 with byte formed by Bit Right Shifter.
21. End Around Right Byte Shifter (AC 3.8) places byte 7 into byte position 2 to form Byte Shift Data (00 00 02 00 00 00 00 00).
22. A/C Stream Memory Request Control (AC 3.6) sends A/C Stream LM Request to LM 3.0.

23. Mark Bit Generator (AC 3.17) sets Mark bit 2.
24. LM System Virtual Address Select Mux (AC 3.15) sends AC Address to Address Mux (LM 3.5).
25. Byte Shift Data byte 2 enters A/C1 Buffer Register (AC 3.9) under control of Mark bit 2.
26. A/C1 Buffer Register sends LM Write Data (00 00 02 00 00 00 00 00) to LM Write Data Register (LM 3.16).
27. Mark Bit Register (AC 3.17) sends AC Mark bit 2 to Mark Bit Generator (LM 3.7).
28. Complement Control (AC 3.14) sets Byte Shift Counter (AC 3.9) to 5 which is complement of Address Adder bits 29 through 31 (byte number 2).
29. Response Translator (LM 3.4) sends A/C Response to AC 3.0. This indicates LM has received data to be stored at addressed CM location.
30. Response Translator sends second A/C Response to AC 3.0. This indicates data at addressed CM location is available to be loaded.
31. LM Read Data Mux (LM 3.16) sends LM Read Data (0---0) to A/C Stream Assembly Mux/Register (AC 3.8) through Pass On (OPI 3.11).
32. LM Read Data passes unchanged through Store Bit/All Other Operations Select Mux (AC 3.8) as Assembly Register data.
33. End Around Right Byte Shifter (AC 3.8) shifts byte 2 into byte position 7 (00 00 00 00 00 00 00 00) according to Byte Shift Count (5).
34. Bit Right Shifter (AC 3.8) shifts byte 7 bit 6 (62) into bit position 7 (63) according to Bit Right Shift Count (6). It does this by first transposing bits (bit 56 becomes 63, 57 becomes 62, and so on) and then right-shifting end-off.
35. Bit Right Shifter sends Final Load Bit [63 (clear)] to A1 Buffer Register (AC 3.9) which is also loaded with zeros in bit positions 0 through 62.
36. A1 Buffer Register sends Load Data (0---0) to Data Interchange Register (OPI 3.12).
37. Data Interchange Register sends X1 Register data (Load Data selected by RDSa field in general micrand) to Register File (OPI 3.5).

#### STORE BYTES INSTRUCTION

The store bytes (A5jk1D) instruction transfers a field of up to eight bytes from the Xk Register to consecutive locations in CM through the A/C Stream in AC. Byte field length is specified by the value of XO Register bits 61 through 63. For lengths less than eight bytes, the instruction right-justifies the bytes in the Xk Register. The transfer may cross a word boundary in CM. If so, the rightmost bytes in the Xk Register transfer first. The Address Adder in AC forms the rightmost CM byte address by adding:

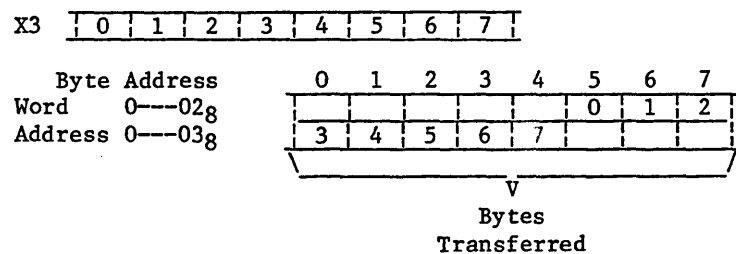
- Byte number field in Aj Register.
- D field in instruction.
- Lower (rightmost) 32 bits in Xi Register.
- Length field in XO Register.

Figure 4-24 shows timing and the following steps describe the sequence of events for a typical 8-byte transfer through the A/C stream. This transfer crosses a word boundary in CM. Parameters for this example are:

$A_j + D + X_i = 0\text{---}025_8$  (leftmost byte address)

Length = 7

Leftmost + Length =  $0\text{---}034_8$  (rightmost byte address)



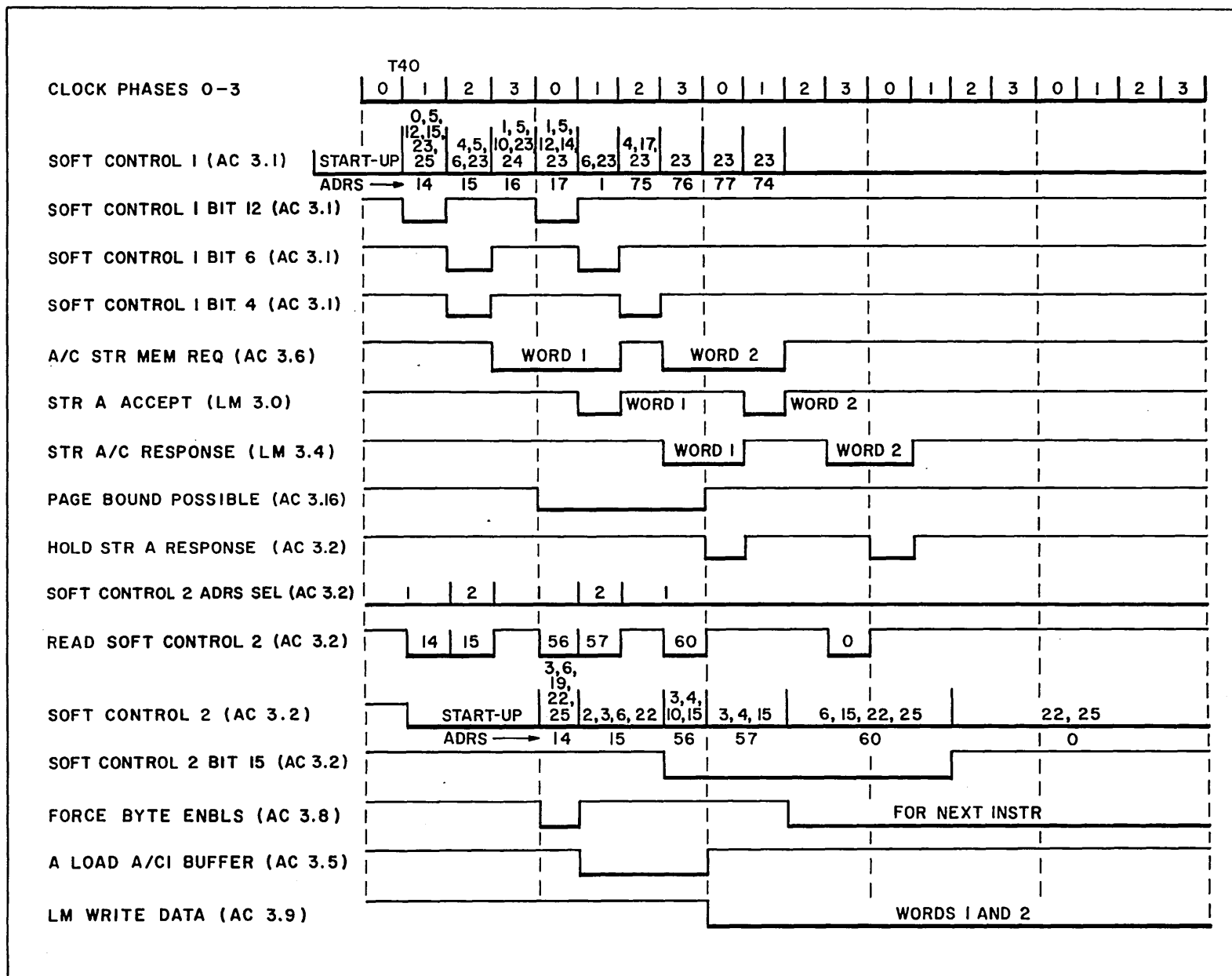


Figure 4-24. Store Bytes Instruction

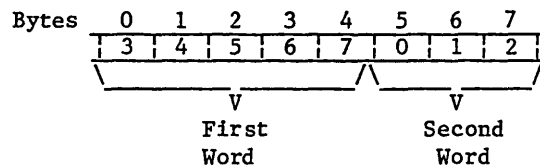
1. Functional Unit Micrand Register (CST 3.5) sends bytes 0 and 1 of Functional Unit Micrand (3860) to AC Micrand Register (AC 3.0).
2. A/C Stream Length Counter (AC 3.13) sets to 0.
3. AC Address Offset Mux (OPI 3.8) sends instruction D field (AC Address Offset selected by IDS field in general micrand) to input A of Address Adder (AC 3.14) through Address Select Mux (AC 3.13). Operand Data Mux (OPI 3.10) sends byte number in Aj Register (A/B Operand selected by RDSa field in general micrand) to input B of Address Adder.
4. Address Adder forms sum of Aj Register byte number and D field.
5. Operand Data Mux (OPI 3.10) sends Xi Register data (A/B Operand selected by RDSb field in general micrand) to input B of Address Adder (AC 3.14). Output of Address Adder feeds back to input A.
6. Address Adder forms sum of Aj Register byte number, D field, and Xi Register data (leftmost byte address 0---025<sub>8</sub>).
7. Address Adder bits 29 through 31 (byte number 5) enter Byte Address Holding Register (AC 3.17).
8. Soft Control 1 (AC 3.1) activates.
9. C Operand Register (OPI 3.11) sends XO Register bits 55 through 63 (length field selected by RDSc field in general micrand) to B Feeder Register (AC 3.14) through Address Offset Mux (AC 3.13).
10. B Feeder Register sends Address Offset to input B of Address Adder (AC 3.14). Output of Address Adder feeds back to input A.
11. Address Adder forms sum of 0---025<sub>8</sub> and length field 0---07<sub>8</sub> (rightmost byte address 0---034<sub>8</sub>).
12. Address Adder generates a carry into bit position 28 as follows. This indicates two CM word locations are required to store these bytes.

Bit Position	<u>27</u>	<u>28</u>	<u>29</u>	<u>30</u>	<u>31</u>
Leftmost Byte Address (25 <sub>8</sub> )	1	0	1	0	1
Length Field (07 <sub>8</sub> )	0	0	1	1	1
Rightmost Byte Address (34 <sub>8</sub> )	1	1	1	0	0

13. Holding Register (AC 3.14) sends Word Boundary Detected to Word Boundary Detector (AC 3.16).
14. Soft Control 2 (AC 3.2) activates and starts word boundary control cycle. If no word boundary, a different cycle would start.
15. Rightmost byte address 0---034<sub>8</sub> enters A/C Stream Address Counter (AC 3.15).
16. Address Adder bits 29 through 31 (byte number 4) enter Byte Shift Counter (AC 3.9).
17. Address Adder bits 29 through 31 (byte number 4) enter Final A/C Stream Byte Number Register (AC 3.9).



18. Mark Bit Generator (AC 3.17) examines rightmost byte address (4) in Final A/C Stream Byte Number Register (AC 3.9) and generates Mark bits for word to be stored at CM word address 0---03<sub>8</sub> (11111000).
19. Byte Shift Counter (AC 3.9) increments from 4 to 5.
20. Operand Data Mux (OPI 3.9) sends X3 Register data from Register File (OPI 3.5) (selected by RDS<sub>c</sub> field in general micrand) to C Operand Register (OPI 3.11).
21. C Operand Register sends X3 data to A/C Stream Assembly/Mux Register (AC 3.8).
22. X3 data passes unchanged through Store Bit/All Other Operations Select Mux (AC 3.8).
23. End Around Right Byte Shifter (AC 3.8) shifts X3 data bytes according to Byte Shift Count (5) to form two CM words.



24. LM System Virtual Address Select Mux (AC 3.15) sends first AC Address to Address Mux (LM 3.5).
25. A/C Stream Memory Request Control (AC 3.6) sends first A/C Stream LM Request to LM 3.0.
26. Byte Shift Data enters A/C1 Buffer Register (AC 3.9) and remains there throughout this instruction.
27. A/C1 Buffer Register sends all eight shifted bytes to LM Write Data Register (LM 3.16).
28. Mark Bit Register (AC 3.17) sends AC Mark bits 0 through 4 to Mark Bit Generator (LM 3.7).
29. Accept Control (LM 3.0) sends first Stream A Accept to AC 3.0.
30. A/C Stream Address Counter (AC 3.15) decrements by eight (0---24<sub>8</sub>).
31. LM System Virtual Address Select Mux (AC 3.15) sends second AC Address to Address Mux (LM 3.5).
32. Response Translator (LM 3.4) sends first Stream A/C Response to AC 3.0.
33. A/C Stream Memory Request Control (AC 3.6) sends second A/C Stream LM Request to LM 3.0.
34. Accept Control (LM 3.0) sends second Stream A Accept to AC 3.0.
35. Mark Bit Generator (AC 3.17) examines leftmost byte address (5) in Byte Address Holding Register (AC 3.17) and generates Mark bits for word to be stored at CM word address 0---02<sub>8</sub> (00000111).

36. A/C1 Buffer Register sends all eight shifted bytes to LM Write Data Register (LM 3.16).
37. Mark Bit Register (AC 3.17) sends AC Mark bits 5 through 7 to Mark Bit Generator (LM 3.7).
38. Response Translator (LM 3.4) sends second Stream A/C Response to AC 3.0.

**PART 3**

**BUSINESS DATA PROCESSOR (BDP)**



This part of this section describes BDP which executes eighteen instructions under micrand control.

- Numeric Instructions
  - Decimal Sum (70jk)
  - Decimal Difference (71jk)
  - Decimal Product (72jk)
  - Decimal Quotient (73jk)
  - Decimal Compare (74jk)
  - Numeric Move (75jk)
  - Decimal Scale (E4jkiD)
  - Decimal Scale Rounded (E5jkiD)
- Byte Instructions
  - Move Bytes (76jk)
  - Byte Compare (77jk)
  - Byte Compare, Collated (E9jkiD)
  - Byte Translate (EBjkiD)
  - Edit (EDjkiD)
  - Byte Scan While Nonmember (F3jkiD)
- Calculate Subscript Instruction (F4jkiD)
- Immediate Data Instructions
  - Move Immediate Data (F9jkiD)
  - Compare Immediate Data (FAjkiD)
  - Add Immediate Data (FBjkiD)

This part describes the major components shown on BDP 1.0 and the data paths for each instruction. Refer to the hardware reference manual for details of each instruction, data types, instruction formats, micro operations, and descriptor formats. This knowledge is critical to understanding the BDP.

#### GENERAL DESCRIPTION

The BDP processes variable length strings of 8-bit bytes that comprise the data fields associated with BDP instructions. BDP receives, processes, and transmits only bytes. BDP is not associated with CM addressing or the assembly/disassembly of CM data words; AC performs these tasks.

#### MICRAND CONTROL

When BDP receives a 64-bit micrand from CST, BDP contains all the information necessary to process data when it arrives. The most one micrand can accomplish in BDP is one input and one output. That is, it can receive and process 0 through 256 bytes of data from one or both source fields, write them into the Buffer RAM, read them from the Buffer RAM, and transmit them to AC. The simpler BDP instructions are handled with one BDP micrand while the more complex instructions require more than one micrand. Although most micrands include both input and output, depending on the function, some micrands process an input only, some an output only, and some process no bytes at all but some other function.

## BUFFER RAM

The Buffer RAM provides a breakpoint for point of no return (PONR). All bytes destined for CM are first stored in the Buffer RAM. When the last byte of data is in the Buffer RAM, the BDP is at the PONR and evaluates and reports to ICC any abnormal conditions detected. ICC then decides whether processing should be continued or aborted. If processing is to continue, the Buffer RAM contents are read and sent to AC. Not all micrands include a PONR. Multimicrand instructions have only one PONR which is always in the last micrand.

## NOTE

BDP always reports conditions when the last byte is stored in the Buffer RAM, regardless of PONR; the exception is BDP Invalid Data, which is reported only on the last micrand.

## BDP BUSY

BDP is busy for as long as necessary to process data. BDP Busy always sets on receipt of a micrand and clears when the last byte of data has been processed within BDP (or sooner on housekeeping micrands).

## INTERFACES

### AC Interface

BDP and AC work together to handle CM data. When Aj source data is involved, AC references CM for an Aj source word, disassembles it in the AC A stream and sends it to BDP A stream. The AC and BDP B streams handle Ak source data the same way. The BDP C stream transmits data processed by BDP to AC C stream where it is assembled into words and stored in CM as Ak destination data.

Prior to data processing, AC also transmits the Aj and/or Ak field lengths to BDP. AC decides whether to use the 8-bit descriptor length or the 9-bit X Register length for a given BDP data field.

### ALN Interface

The BDP Binary/Decimal Converter and the ALN Multiply Network convert decimal bytes to binary words. This conversion is needed for the numeric move instruction, calculate subscript instruction, and decimal multiply/divide instructions (which also use ALN to form a binary product/quotient).

For numeric move instructions requiring a conversion, ALN transmits the binary result back to BDP through the Register File (OPI) and AC. BDP then stores the binary result in CM through AC.

For multiply/divide, ALN converts both decimal input operands and stores the binary results separately in the Register File. ALN then forms the product/quotient which it stores in the Register File, which then transmits the product/quotient back to BDP through AC. BDP converts to decimal and stores the result in CM through AC.

In converting decimal to binary, BDP converts each decimal byte to its binary equivalent and transmits the byte to ALN which adds it to a cumulative result. ALN then multiplies the cumulative result by 100 (decimal). This process repeats until all source bytes are processed.

#### OPI Interface

The X0, X1 Result Selector partially forms the X0 and X1 Register results which are transmitted to OPI during certain instructions, that is, compare status.

The BDP X0 Result provides a copy of the last Buffer RAM address. This is a sequence count used on compare instructions. BDP X0 Result may also contain the Ak Descriptor Length Field used on the byte scan instruction.

The BDP X1 Result bits 32 and 33 represent various compare status flags. Bits 56 through 63 contain a copy of the last byte processed during a byte scan instruction.

#### MAC Interface

BDP contains 6 random access memories (RAMs) that function as read only memories (ROMs).

- A Stream Decode.
- B Stream Decode.
- Instruction Specification Error.
- Translate Encode.
- Convert to Binary/Decimal.
- Converter Multiply by 256.

MAC preloads these RAMs during system initialization. Refer to BDP Memory Write in section 2 for the loading sequence. Refer to Maintenance Aids in the Maintenance and Parts Data Manual for maps of BDP memories.

BDP parity checkers send parity error signals to Processor Fault Status (PFS) Registers 80, 81, and 82 in MAC. The Processor Test Mode (PTM) Register in MAC sends various test signals to BDP.

#### CST Interface

CST sends BDP a 64-bit Functional Unit Micrand which controls data processing within BDP. The BDP Condition Bit Select Mux sends status signals to CST. These BDP Condition Bits control CST addressing for the next micrand.

### ICP Interface

ICP sends BDP Go to initiate a BDP operation. ICP Rank 32 sends the BDP Descriptor Type Field which specifies the type of data to be processed by BDP.

### ICC Interface

The Processor Detected Malfuction (PDM) and Exception Detectors send various status signals to ICC. The Exception Detector senses arithmetic overflow and loss of significance, divide fault, instruction specification error, and invalid data. The PDM Detector senses parity errors before and after PONR.

Although BDP continuously checks for internal parity errors and sends status signals to PFS Registers in MAC, the PDM signals are sent to ICC only when BDP is busy. Parity errors detected before PONR can be retried, those detected after PONR cannot be retried. Generally, for a micrand with a PONR (Micrand Bit 59 set), all parity errors except those detected on a C stream operation generate BDP PDM Before PONR, and those occurring during a C stream operation generate BDP PDM After PONR. For a micrand with no PONR (Micrand Bit 59 clear), all parity errors generate BDP PDM Before PONR. Two parity errors generate no PDM signal because the data is propagated to other functional units. These are Translate Encode RAM (data to AC) and Convert to Binary/Decimal RAM (data to ALN) parity errors.

### DESCRIPTORS

On a given BDP instruction, one or both of the Aj/Ak descriptors is latched in BDP and processed. The descriptor in BDP consists only of the data type and length fields. Regardless of any other use, a descriptor is checked for data type and length violations which constitute an instruction specification error. This results in aborting the micrand in process. Further, unless overridden by micrand command, the descriptor data types control the actions of the A and B input streams and the C output stream. The Ak length field is used in three other ways.

- When zero, conditions no operation.
- Compares with Buffer RAM Address Counter to determine when destination field length is satisfied.
- Enters XO Register on no hit condition during scan instruction.

### INSTRUCTION SPECIFICATION ERROR

The Instruction Specification Error RAM is always referenced after receiving either descriptor. This RAM contains bits 22 through 31 which evaluate the data type and length. Bit 22 is a single bit specifying a bad data type (set) or a good type (clear). Bits 23 through 31 represent the maximum allowable length and are compared against the 9-bit descriptor length.



## SCALE CONTROL

Scale Control contains a counter that holds one of two quantities: the immediate byte used for immediate data instructions or the 8-bit shift count used on Scale and Scale-Rounded instructions. In either case, just prior to the micrand that uses the data, AC transmits the byte on the B stream.

If an immediate data instruction, the counter functions as a holding register, providing the Immediate Operand to A Stream Stage 1.

If a scale instruction, the counter decrements on a left shift (shift-count positive) and increments on a right shift (shift-count negative). A shift is complete when the shift count reaches zero. At zero, the Shift Left or Shift Right signal drops.

A shift-left operation creates a zero digit for each shift count until the shift is complete, then allows the A<sub>j</sub> source data to be processed. Shift Left activates Pause Control to prevent A<sub>j</sub> source data from leaving AC. Shift Left creates zero bytes in Common Stage 5 by faking an A Stream Stage 3 active signal. This forces A Stream Stage 4 to activate, resulting in zeros written into the Buffer RAM.

A scale-right instruction ignores as many least significant digits, as represented by the shift count, by not writing them into the Buffer RAM. This is done by ignoring the A Stream Stage 3 active signal which would normally cause Pause A, B Stream to drop and pass the corresponding digit from A Stream Stage 4 to Common Stage 5. Shift Right activates if a digit is ready in A Stream Stage 3 and if the shift count is not zero.

A scale-rounded instruction senses for A stream digit value greater than four, then adding plus one to the next digit if the next digit is the first (least significant) after scaling is complete. A carry is propagated, if necessary.

## PAUSE CONTROL

Pause Control generates Pause A, B Stream signals that command AC to stop moving data on a particular stream (hold it in place) until the signal drops.

Pause Control causes A (B) Stream Stages 1 and 2 and most of AC A (B) stream to remain as is while the byte in stage 2 is unpacking the first of two digits into stage 3. The pause then drops as the second digit is gated into stage 3.

Pause Control also enables stream synchronization. If a source field byte is in one input stream stage 4 without a corresponding byte in the other stage 4, the first stream pauses until the other byte arrives.

There are many other reasons to pause a stream, such as shift left, wait for A<sub>k</sub> descriptor, and so forth.

## DATA PATHS

The following paragraphs and BDP 1.0 describe the data paths through BDP.

### A AND B STREAM STAGES 1 THROUGH 4

The A and B streams are each composed of four stages. Stages 1 through 3 evaluate and manipulate the incoming data according to the Aj (A stream) or Ak (B stream) Descriptor Type Field.

### Decimal (Types 0 through 8)

- Decode data to a string of bytes containing a digit in the lower bits and zeros in the upper.
- If packed, unpack.
- If signed, record sign.
- If separate sign, strip it off.
- If slack digit, zero it.
- Fill with 00 (hexadecimal), if necessary.
- Validate each byte according to descriptor data type.

### Alphanumeric (Type 9)

- Pass on data intact with 20- (hexadecimal) fill byte, if necessary.

### Binary (Type 10, 11)

- Pass on data intact.
- If signed, record sign.
- Fill (sign-extend) with 00 or FF (hexadecimal), if necessary.

### Translate (Types 12 through 15)

- Convert to normal type 2, 3, 10, or 11 before manipulating data (refer to descriptions for types 2, 3, 10, and 11).

#### Stage 4

If necessary, stage 4 synchronizes the A and B streams. Any byte active in stage 4 that doesn't have a corresponding byte in the opposite stage 4 remains there until the corresponding byte appears. Synchronizing decisions are made based on stage 3 status.

#### ALU AND COMMON STAGES 5 THROUGH 7

Stage 5 is the first stage of the combined or common input stream. The Decimal/Binary ALU is between stages 4 and 5. This ALU adds, subtracts, or does nothing depending on the instruction. Actually, the ALU is always in add mode. The A stream input may be complemented to cause a subtract. The add may be in binary or decimal. Either input may be forced to zero which allows the other input to pass unchanged. Stage 6 repacks decimal data (if necessary) and adds a sign digit/byte (if necessary). Stage 7 stores data from Common Stage 6, Edit, or the Register Files in the Buffer RAM.

#### BUFFER RAM

The Buffer RAM holds all bytes processed as input data or from Edit, and has a 256-byte capacity which can hold the maximum destination length.

When writing processed bytes, the Buffer RAM Address Counter always starts at address zero. Also in most cases when outputting from the Buffer RAM, the Address Counter starts at address zero, that is, left-to-right input results in left-to-right output and right-to-left input results in right-to-left output. The exception is data written into the Buffer RAM to be converted. In this case, the write starts at address zero and the address increments; the read begins at the address of the most significant nonzero byte and decrements to zero. Leading zeros are truncated by a circuit that records the Buffer RAM address of the most significant nonzero byte [or the address of the most significant non-FF (hexadecimal) byte in the case of signed negative binary]. After writing into the Buffer RAM is complete, this read address is gated into the Address Counter.

Any time Common Stage 7 is active, a Buffer RAM write occurs, even if the instruction specifies input data only, that is, a compare and the Buffer RAM data is not used.

The Address Counter compares with Ak Length Minus One to determine whether a write into the Buffer RAM is beyond the destination field length. This is useful primarily for halting an input and for determining overflow.

Various conditions active in Common Stage 7 and selected by the micrand determine when input is complete. When complete, a stop timing chain activates to halt the input, report on PONR and status, and begin the output of data from the Buffer RAM. If input only, the micrand blocks the output. If output only, the micrand activates the stop timing chain immediately. If neither input nor output, the micrand blocks the output immediately.

#### C STREAM STAGES 1 THROUGH 5

The C Stream Stages 1 through 5 further process data read from the Buffer RAM and transmits it usually to AC, but also to ALN or the BDP Binary/Decimal Converter if a convert. If to AC, only the number of bytes in the destination length are transmitted. If a convert, then only the significant bytes are transmitted.

Stage 1 latches data read from the Buffer RAM. The Complement and Combined Sign Adders are between stages 1 and 2. The Complement Adder performs decimal recomplement, or a one's or two's binary complement. This further processes data that was impossible on the A and B streams primarily because the entire source field contents had to be known first. The Combined Sign Adder forms the sign digit(s) for the applicable data types.

Stage 3 encodes decimal data. The Buffer RAM held all digits and the digit/byte reserved for signs, but not the final form. Signs, American Standard Code for Information Interchange (ASCII) bias, and slack digits are inserted here.

The PONR occurs when the last byte is written into the Buffer RAM. All abnormal conditions are reported at that time. However, further processing takes place in output stages, thereby modifying data after the PONR. To handle this, BDP latches various conditions during the write. At PONR time, this allows BDP to anticipate accurately the final destination data values and report accurately on conditions. Stage 4 data addresses the Translate Encode RAM. If the output is specified as a translate data type, the RAM data is selected into stage 5. Otherwise, stage 4 data is selected into stage 5. Stage 5 is the C stream interface to AC.

#### BINARY/DECIMAL CONVERTER

If data is destined for ALN for convert to binary or to the Binary/Decimal Converter for convert to decimal, the data byte addresses the Convert to Binary/Decimal RAM. This RAM contains both the binary and decimal equivalent of the data byte. The appropriate value goes on to the convert in process. This RAM (stage 5) is the interface to ALN and to the Decimal Converter.

The Binary/Decimal Converter completes the conversion of binary bytes to a string of decimal bytes by adding and multiplying.

During the micrand that performs the decimal convert, the A stream processes the binary data from AC and stores it into the Buffer RAM. The Buffer RAM is then read byte-by-byte, converted into decimal by the Convert to Binary/Decimal RAM, added to a cumulative result by the Converter Decimal Adder, and stored in the Converter Result RAM.

During the micrand that stores the decimal result, the Converter Result RAM transmits Converted Decimal to A Stream Stage 1 as unpacked decimal. This is written into the Buffer RAM after being reformatted to the destination data type, then transmitted to AC.

The convert micrand initializes the Binary/Decimal Converter RAMs to zero at the same time it processes the input data. As each byte is read from the Buffer RAM, the Convert to Binary/Decimal RAM converts eight binary bits to ten decimal bits. The Converter Decimal Adder adds the converted bits to the cumulative result. The cumulative result is then multiplied by 256 (decimal) by passing each byte from the result through the Converter Multiply by 256 RAM and adding this 18-bit Multiply Product to the cumulative result.

#### REGISTER FILES A AND B

Register Files A and B hold data specified in certain instructions as the  $A_i$  plus d tables. They are loaded by special micrands prior to the micrands that use the data. This provides a fast and simple way of referencing the tables.

To load  $A_i$  plus  $d$  source data into Register Files A and B, A stream input data is written into the files using an Address Counter to increment the address starting at zero. Both files are written with the same data with the exception of loading the edit mask which is written only into Register File B.

When using an  $A_i$  plus  $d$  table, A and/or B stream input data address the respective files. This provides substitute data for processing collated compare, byte translate, and byte scan instructions. Collated compare and byte scan instructions sense the data for compare equal or scan hit conditions and halt operation. For the byte translate instruction, Register File A Data enters Common Stage 7 as destination data.

Edit uses the files differently. Prior to the micrand that edits, the edit mask enters Register File A and the  $A_j$  source data (the original field if type 9 or a string of unpacked decimal digits devoid of sign and slack if decimal) enters Register File B. When editing, the Address Counter for each file increments each time a new value is needed.

#### Register File Compare Control

Compare instructions use Register File Compare Control to determine results. It determines that the value of A stream data is equal to, less than, or greater than B stream data.

#### Scan Hit Control

Scan instructions use Scan Hit Control to determine if a specified byte(s) exists in the  $A_k$  source field. If a source field byte has a corresponding bit set in Register File B RAM, a Scan Hit occurs. This indicates a specified byte has been detected in the source field.

#### EDIT

The Edit network executes the edit instruction. Register File A is preloaded with the edit mask. Register File B is preloaded with the  $A_j$  source data. The edit instruction formats the source data under control of the edit mask which consists of a sequence of micro operations. The edit mask consists of a string of up to 256 8-bit bytes. Each byte contains either a 4-bit micro operation (MOP) code and a 4-bit specification value (SV) or a data byte to be inserted in the final result. Edit Control decodes these values which control execution of the edit instruction. Certain MOPs and SVs address the Edit Special Characters Table (SCT) and Symbol Mask RAMs which contain characters and symbols needed by the edit instruction. The Edit Result Mux produces the Edit Result which is stored in the Buffer RAM.

#### INSTRUCTION SEQUENCES

The following paragraphs and BDP 1.0 describe the sequence of events within BDP for initialization and for each BDP instruction. Each description is preceded by a graphic illustration (figures 4-25 through 4-43) of the microcode sequence which controls data flow within BDP, AC, and ALN. The keys to these illustrations are listed as follows.

- Time advances from left to right. Micrand lengths shown are typical but otherwise irrelevant.
- BDP #(XX) is the BDP feature referenced by microcode. This number can be found in the microcode listing.

- AC features are for reference only. They are:

ACA Sends Aj descriptor and data to BDP A stream.  
ACB Sends Ak descriptor and data to BDP B stream.  
ACC Receives BDP data on C stream.  
ACI Sends immediate byte to BDP B stream.  
ACAI Sends Ai plus d data to BDP A stream.  
ACKD Sends Ak descriptor to BDP B stream.  
ACP Prevalidates memory addresses for data in C stream field.  
ACAR Sends Register File data to BDP A stream.

- ALN features are for reference only. They are:

ALNRFW Writes data into Register File.  
ALNXOW Writes data into XO Register.  
ALNX1W Writes data into X1 Register.  
ALNC9 Processes first nine bytes of decimal-to-binary conversion.  
ALNC1 Processes single byte of decimal-to-binary conversion.

- CPU point of no return (PONR) is shown.

# BDP INITIALIZATION SEQUENCE

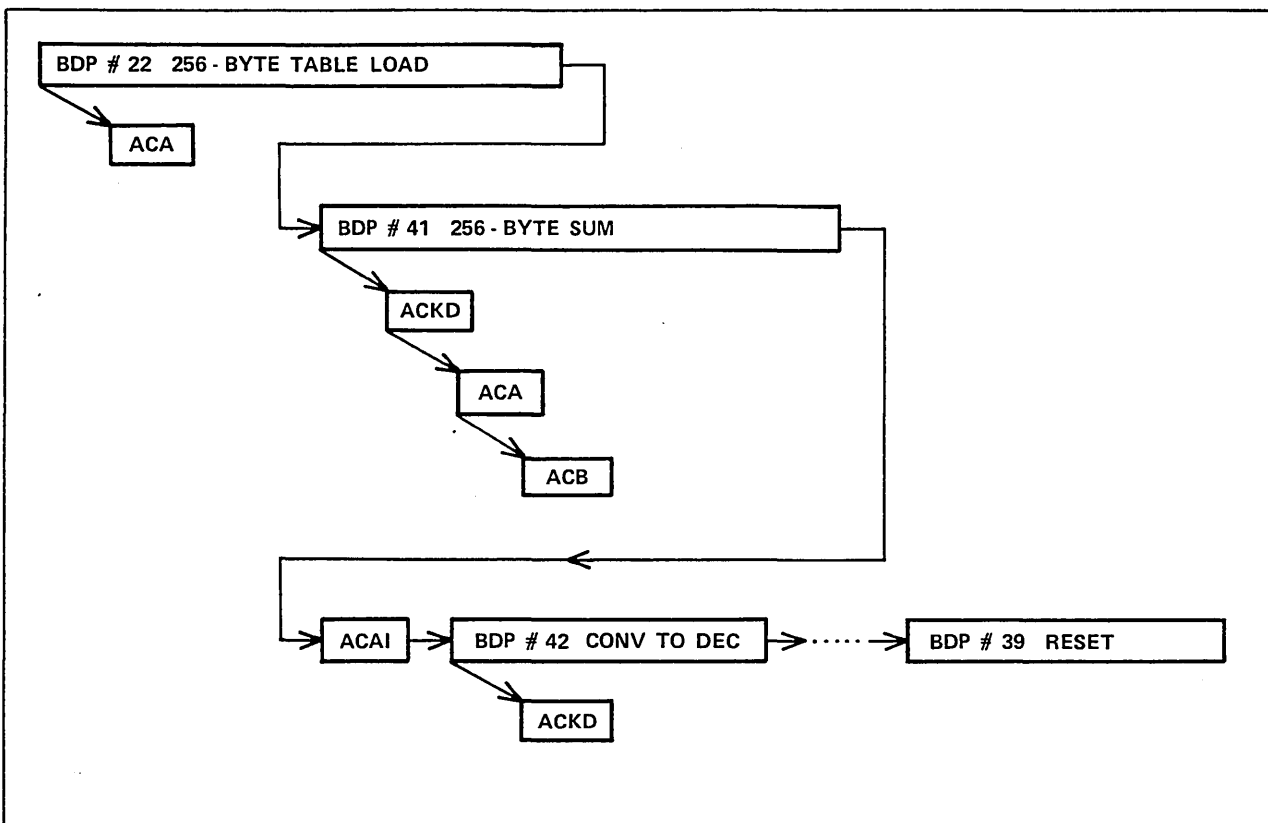


Figure 4-25. BDP Initialization Micrands

This operation is part of CP initialization. This sequence prepares BDP for use by clearing all parity errors. It accomplishes this by passing data with good parity through all data paths and storing this data into all locations of Register Files A and B, Buffer, and Convert Result RAMs. Four BDP micrands cause the following data flow.

## First Micrand (#22)

This micrand writes good data into all 256 locations of Register File A and B RAMs, and clears the A stream.

1. AC A Stream Disassembly Mux sends eight bytes of zeros to BDP A Stream Stages 1 through 3.
2. A Stream Stage 3 sends eight bytes of zeros and 248 bytes of blanks (20 hexadecimal) to Register File A Stream Stage 4.
3. Register File A Stream Stage 4 writes 256 bytes of zeros and blanks into Register File A and B RAMs.

#### Second Micrand (#41)

This micrand writes good data into all 256 locations of the Buffer Ram, and clears the B and C streams.

1. AC sends Descriptor Length Field of 256 to BDP Ak Length Register.
2. AC A Stream Disassembly Mux sends eight bytes of zeros to BDP A Stream Stages 1 through 3.
3. AC B Stream Disassembly Mux sends eight bytes of zeros to BDP B Stream Stages 1 through 3.
4. A and B Stream Stage 3 each send 256 bytes to stage 4.
5. A and B Stream Stage 4 synchronize 256 byte pairs of zeros.
6. Decimal/Binary ALU adds 256 byte pairs of zeros.
7. Summed 256 byte pairs pass through Common Stages 5 and 6.
8. Common Stage 7 stores 256 bytes in Buffer RAM.
9. Buffer RAM sends 256 bytes to AC through C Stream Stages 1 through 5. AC ignores this data.

#### Third Micrand (#42)

This micrand writes good data into all 32 locations of the Convert Result RAM, clears the convert data paths, and clears the immediate byte holding register (Scale Counter in Scale Control).

1. Binary/Decimal Converter clears all locations of Converter Result RAM.
2. AC sends immediate byte of zeros over B Stream Data path to Scale Control.
3. AC sends Descriptor Length Field of zero to BDP Ak Length Register.
4. A Stream Stage 3 sends a byte of zeros to A Stream Stage 4 as immediate data.
5. Byte of zeros passes through Decimal/Binary ALU and Common Stages 5 and 6.
6. Common Stage 7 writes byte of zeros into Buffer RAM.
7. Buffer RAM sends byte of zeros to C Stream Stages 1 through 3.
8. C Stream Stage 3 sends byte of zeros to Binary/Decimal Converter which converts byte of zeros to decimal. This clears Binary/Decimal Converter data paths.

#### Fourth Micrand (#39)

This micrand ensures that BDP Busy has cleared after the last micrand. BDP Busy clears after CST issues this micrand.



## NUMERIC INSTRUCTION SEQUENCES

### Decimal Sum (70jk)

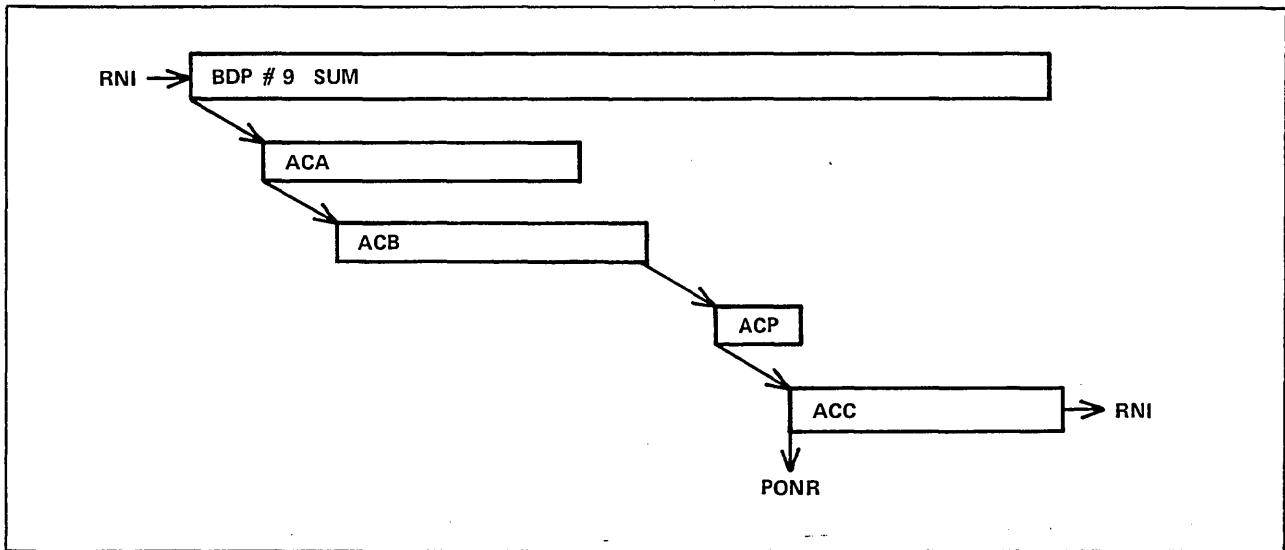


Figure 4-26. Decimal Sum Micrands

This instruction adds decimal data in the Ak source field to decimal data in the Aj source field and places the sum in the Ak destination field. One BDP micrand causes the following data flow.

1. AC A Stream Disassembly Mux sends Aj source decimal byte to BDP A Stream Stages 1 through 3.
2. AC B Stream Disassembly Mux sends Ak source decimal byte to BDP B Stream Stages 1 through 3.
3. A and B Stream Stage 4 synchronize resultant unpacked and unsigned decimal bytes.
4. Decimal/Binary ALU adds or subtracts decimal bytes, depending on Aj and Ak source field signs. If signs are alike, ALU adds and destination field receives same sign. If signs are unlike, ALU subtracts Aj source from Ak source and destination field receives same sign as Ak source. If Aj source value is larger than Ak source value, an erroneous result occurs. However, this error is corrected later (step 9).
5. Common Stages 5 and 6 reformat decimal byte to required destination field data type.
6. Common Stage 7 stores decimal byte in Buffer RAM.
7. Steps 1 through 6 repeat until both source fields are exhausted.
8. Buffer RAM sends decimal byte to C Stream Stages 1 through 5.

9. C Stream Stages 1 and 2 correct erroneous result that occurred if Aj source value was larger than Ak source value on a subtract operation (step 4). The result is corrected by performing a nine's complement plus one to the decimal result and by changing the Ak destination sign to the opposite of the Ak source sign.
10. C Stream Stage 3 encodes decimal result to destination decimal type by forming ASCII bias, digit/byte sign, and slack digit.
11. C Stream Stages 4 and 5 encode translated decimal byte (if necessary) and send Ak destination decimal byte to AC C Stream Assembly Mux.
12. Steps 8 through 11 repeat until Buffer RAM is empty.

#### Decimal Difference (71jk)

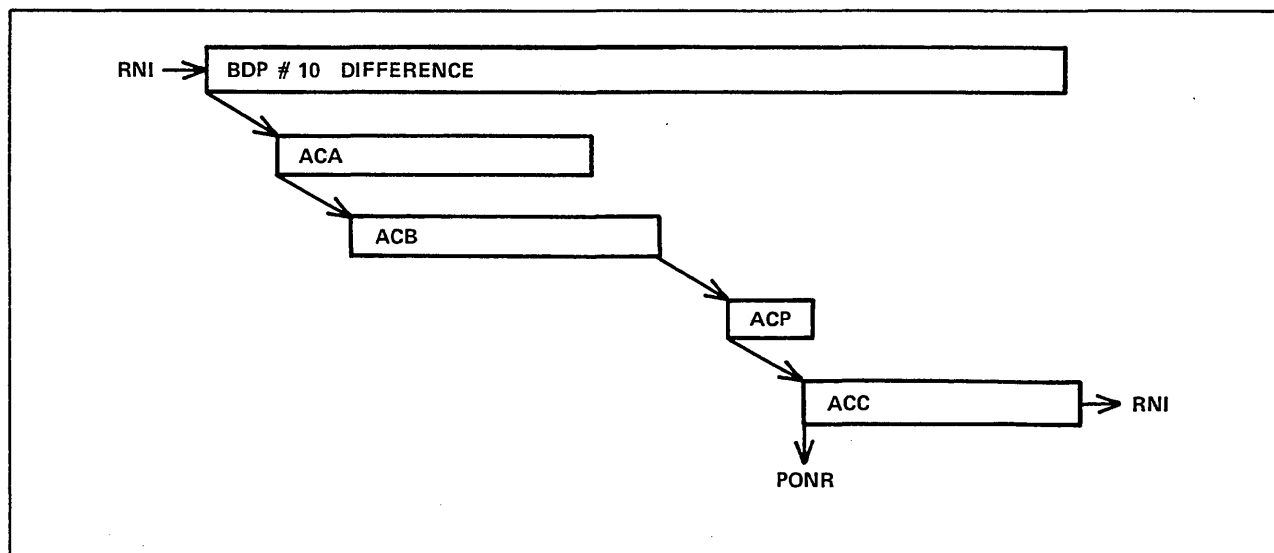


Figure 4-27. Decimal Difference Micrands

This instruction subtracts decimal data in the Ak source field from decimal data in the Aj source field and places the difference in the Ak destination field.

This instruction is the same as the decimal sum (70jk) instruction except source signs affect the arithmetic differently. If signs are alike, the ALU subtracts Aj source from Ak source and destination field receives same sign as Ak source. If Aj source is larger than Ak source, an erroneous result occurs which is later corrected in C Stream Stages 1 through 3. If signs are unlike, ALU adds and destination field receives same sign as Ak source.

# Decimal Product (72jk)

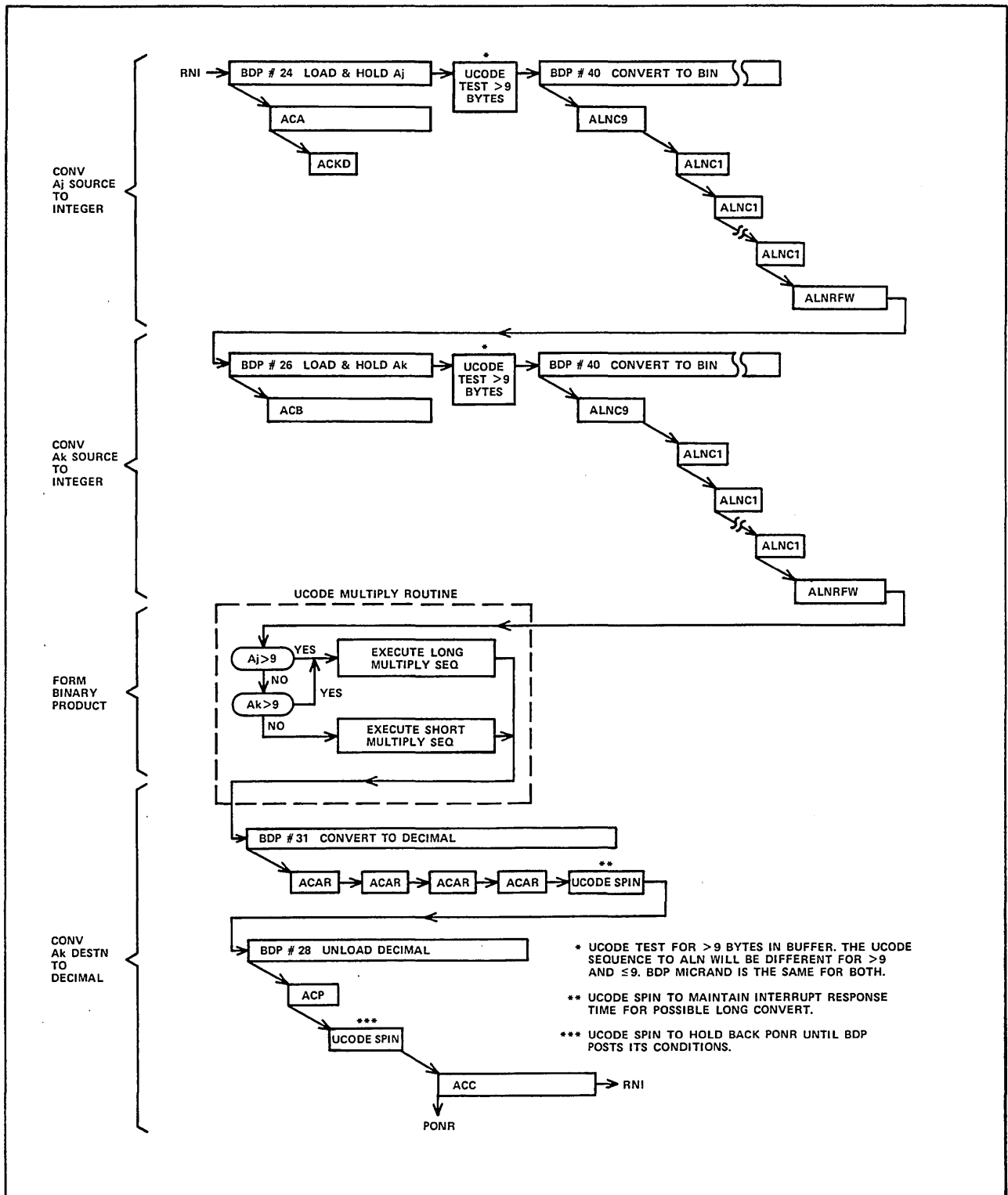


Figure 4-28. Decimal Product Micrand

This instruction multiplies decimal data in the Ak source field times decimal data in the Aj source field and places the product in the Ak destination field. Six BDP micrands cause the following data flow.

#### First Micrand (#24)

1. AC A Stream Disassembly Mux sends Aj source decimal byte to BDP A Stream Stages 1 through 3.
2. Resultant unpacked and unsigned decimal byte passes through A Stream Stage 4 and Decimal/Binary ALU.
3. Common Stages 5 and 6 reformat byte to packed decimal.
4. Common Stage 7 stores packed decimal byte in Buffer RAM.
5. Steps 1 through 4 repeat until Aj source field is exhausted.
6. BDP Condition Bit Select Mux sends BDP Condition (eight or less or more than nine significant bytes in Buffer RAM) to CST. This allows CST to issue appropriate microcode sequence for convert to binary.

#### Second Micrand (#40)

1. Buffer RAM sends packed decimal byte to C Stream Stages 1 through 3.
2. C Stream Stage 3 sends packed decimal byte to address Convert to Decimal/Binary RAM.
3. Convert to Decimal/Binary RAM sends BDP Converted Binary byte to ALN.
4. ALN completes decimal to binary conversion as described later under Convert to Binary Sequence.
5. Steps 1 through 4 repeat until Buffer RAM is empty.
6. ALN sends binary equivalent of decimal data to Register File locations 30 and 31.

#### Third Micrand (#26)

This sequence is the same as the first micrand except Ak source data loads into the Buffer RAM through the B Stream.

#### Fourth Micrand (#40)

This sequence is the same as the second micrand except ALN sends binary equivalent of decimal data to Register File locations 08 and 09.

#### Microcoded Binary Multiply

1. Register File sends binary equivalent of Aj and Ak fields to ALN.
2. ALN multiplies Ak source field times Aj source field. If both fields were nine or less bytes long, ALN performs a double-precision multiply. If one or both fields are longer than nine bytes, ALN converts both fields to floating-point format, performs floating-point multiply, and converts product back to binary format.
3. ALN sends binary product to Register File locations 3C, 3D, 3E, and 3F.

#### Fifth Micrand (#31)

1. Register File sends 64-bit unsigned binary product to AC A Stream Disassembly network.
2. AC A Stream Disassembly Mux sends unsigned binary product byte to BDP A Stream Stages 1 through 3.
3. Unsigned binary product byte passes through A Stream Stage 4, Decimal/Binary ALU, and Common Stages 5 and 6.
4. Common Stage 7 stores unsigned binary product byte in Buffer RAM.
5. Steps 1 through 4 repeat four times which allows entire unsigned binary product to be stored in Buffer RAM.

#### NOTE

Even though binary product may be 256 bits long (quadruple precision) and no BDP decimal data type can be more than double precision (128 bits), all 256 bits of binary are sent to BDP for conversion to decimal. Convert algorithm converts most significant bytes first. Therefore, entire product must be converted if resultant lower half is to be accurate.

6. Buffer RAM sends binary product byte to C Stream Stages 1 through 3.
7. C Stream Stage 3 sends binary product byte to Binary/Decimal Converter.
8. Binary/Decimal Converter performs binary to decimal conversion as described later under Convert to Decimal Sequence.
9. Steps 6 through 8 repeat until Buffer RAM is empty and entire decimal product is stored in Converter Result RAM.

Sixth Micrand (#28)

1. Convert Result RAM sends Converted Decimal byte (least significant byte first) to A Stream Stages 1 through 3.

NOTE

Convert data is packed decimal and the A Stream is expecting unpacked decimal. Therefore, each convert byte is transmitted twice: first with digits transposed and then normally. This transmission continues until all significant convert RAM data is sent. The last digit is accompanied by last byte status.

2. Converted decimal byte passes through A Stream Stage 4 and Decimal/Binary ALU.
3. Common Stages 5 and 6 reformat decimal byte to required destination field data type.
4. Common Stage 7 stores decimal byte in Buffer RAM.
5. Steps 1 through 4 repeat until destination field length is satisfied.
6. Buffer RAM sends decimal byte to C Stream Stages 1 through 5.
7. C Stream Stage 3 encodes decimal result to destination data type by forming ASCII bias, digit/byte sign, and slack digit.
8. C Stream Stages 4 and 5 encode translated decimal byte (if necessary) and send Ak destination decimal byte to AC C Stream Assembly Mux.
9. Steps 6 through 8 repeat until Buffer RAM is empty.

# Decimal Quotient (73jk)

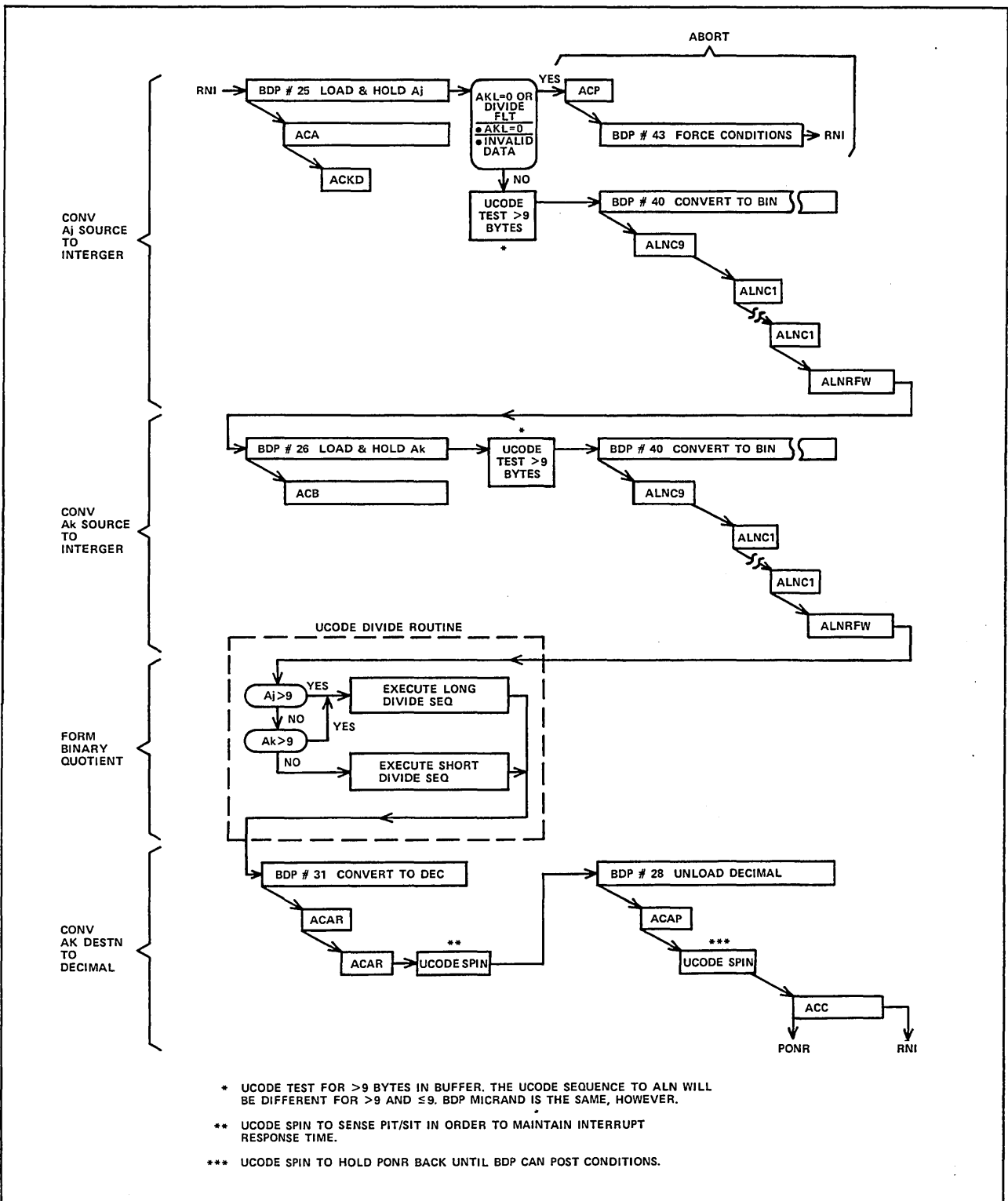


Figure 4-29. Decimal Quotient Micrands

This instruction divides decimal data in the Ak source field by decimal data in the Aj source field and places the quotient in the Ak destination field. Six BDP micrands cause the following data flow.

#### First Micrand (#25)

1. AC A Stream Disassembly Mux sends Aj source data byte to BDP A Stream Stages 1 through 3.
2. Resultant unpacked and unsigned decimal byte passes through A Stream Stage 4 and Decimal/Binary ALU.
3. Common Stages 5 and 6 reformat data to packed decimal.
4. Common Stage 7 stores packed decimal byte in Buffer RAM.
5. Steps 1 through 4 repeat until Aj source field is exhausted.
6. BDP Condition Bit Select Mux sends BDP Condition (eight or less or more than nine significant bytes in Buffer RAM) to CST. This allows CST to issue appropriate microcode sequence for convert to binary.

#### NOTE

Abort divide occurs if Ak length equals zero or a divide fault is sensed. A divide fault occurs if Aj source equals zero, Ak length is not equal to zero, and data type is valid. Micrand #43 exists solely to force BDP to report BDP Invalid Data condition since the first micrand disallowed it.

#### Second Micrand (#40)

1. Buffer RAM sends packed decimal byte to C Stream Stages 1 through 3.
2. C Stream Stage 3 sends packed decimal byte to address Convert to Decimal/Binary RAM.
3. Convert to Decimal/Binary RAM sends BDP Converted Binary byte to ALN.
4. ALN completes decimal to binary conversion as described later under Convert to Binary Sequence.
5. Steps 1 through 4 repeat until Buffer RAM is empty.
6. ALN sends binary equivalent of decimal data to Register File locations 34 and 35.

#### Third Micrand (#26)

This sequence is the same as the first micrand except Ak source data loads into the Buffer RAM through the B stream and abort divide condition is not sensed.



#### Fourth Micrand (#40)

This sequence is the same as the second micrand except ALN sends binary equivalent of decimal data to Register File locations 30 and 31.

#### Microcoded Binary Divide

1. Register File sends binary equivalent of Aj and Ak source fields to ALN.
2. ALN divide Ak source field by Aj source field using double precision.
3. ALN sends binary quotient to Register File locations 3C and 3D.

#### Fifth Micrand (#31)

1. Register File sends 64-bit unsigned binary quotient to AC A Stream Disassembly network.
2. AC A Stream Disassembly Mux sends unsigned binary quotient byte to BDP A Stream Stages 1 through 3.
3. Unsigned binary quotient byte passes through A Stream Stage 4, Decimal/Binary ALU, and Common Stages 5 and 6.
4. Common Stage 7 stores unsigned binary quotient byte in Buffer RAM.
5. Steps 1 through 4 repeat until the entire unsigned binary quotient is stored in Buffer RAM.

#### NOTE

Register File sends two 64-bit words to AC.  
First, from location 3C for least significant half of quotient. Second, from location 3D for most significant half.

6. Buffer RAM sends binary quotient byte to C Stream Stages 1 through 3.
7. C Stream Stage 3 sends binary quotient byte to Binary/Decimal Converter.
8. Binary/Decimal Converter performs binary to decimal conversion as described later under Convert to Decimal Sequence.
9. Steps 6 through 8 repeat until Buffer RAM is empty and entire decimal quotient is stored in Converter Result RAM.

#### Sixth Micrand (#28)

1. Convert Result RAM sends Converted Decimal byte (least significant byte first) to A Stream Stages 1 through 3.

#### NOTE

Convert data is packed decimal and the A Stream is expecting unpacked decimal. Therefore, each convert byte is transmitted twice: first with digits transposed and then normally. This transmission continues until all significant Converter Result RAM data is sent. The last digit is accompanied by last byte status.

2. Converted decimal byte passes through A Stream Stage 4 and Decimal/Binary ALU.
3. Common Stages 5 and 6 reformat decimal byte to required destination field data type.
4. Common Stage 7 stores decimal byte in Buffer RAM.
5. Steps 1 through 4 repeat until destination field length is satisfied.
6. Buffer RAM sends decimal byte to C Stream Stages 1 through 5.
7. C Stream Stage 3 encodes decimal result to destination data type by forming ASCII bias, digit/byte sign, and slack digit.
8. C Stream Stages 4 and 5 encode translated decimal byte (if necessary) and send Ak destination decimal byte to AC C Stream Assembly Mux.
9. Steps 6 through 8 repeat until Buffer RAM is empty.

#### Decimal Compare (74jk)

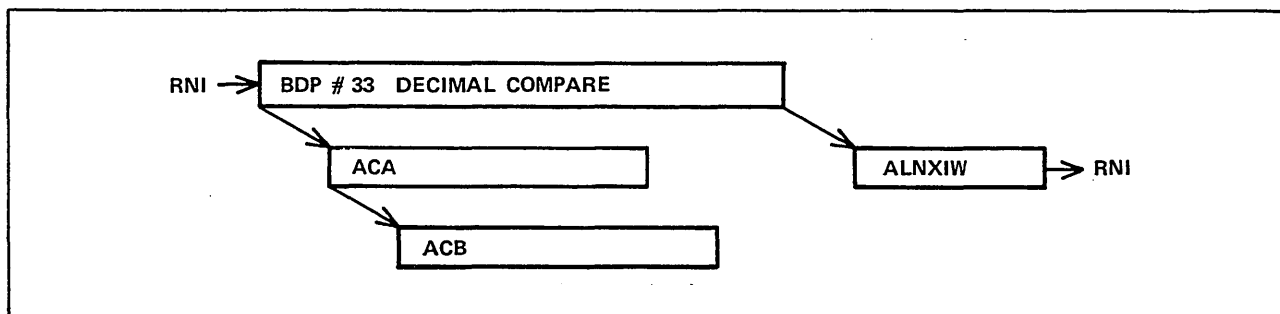


Figure 4-30. Decimal Compare Micrands

This instruction compares decimal data in the Ak and Aj source fields and reports status to the X1 Register. One BDP micrand causes the following data flow.

1. AC A Stream Disassembly Mux sends Aj source decimal byte to BDP A Stream Stages 1 through 3.
2. AC B Stream Disassembly Mux sends Ak source decimal byte to BDP B Stream Stages 1 through 3.
3. A and B Stream Stage 4 synchronize resultant unpacked and unsigned decimal bytes.
4. Decimal/Binary ALU compares decimal bytes (digit by digit) from A and B Stream Stage 4.
5. Common Stage 5 sends comparison signals (Carry Out, Equals Zero) to Register File Compare Control.
6. Steps 1 through 5 repeat until both source fields are exhausted. Shorter field is left-filled with zeros.
7. Register File Compare Control determines final result.
8. X1 Result Selector sends BDP X1 Result to OPI as follows.

<u>Compare</u>	<u>X1 Result</u>			
	<u>32</u>	<u>33</u>	<u>34</u>	<u>63</u>
A = B	0	0	0	— 0
A > B	0	1	0	— 0
A < B	1	1	0	— 0

#### NOTE

Although comparison of two fields is easier and faster if done left to right, decimal fields are compared right to left in order to align the values at the decimal point which is to the right of the rightmost digit. Hence, A > B and B > A FFs remember which type of miscompare was last detected as both fields are compared in their entirety. When comparison is complete, one or neither mutually-exclusive FF indicates the relationship of the decimal values. The two source signs are used to resolve the compare status.

# Numeric Move (75jk)

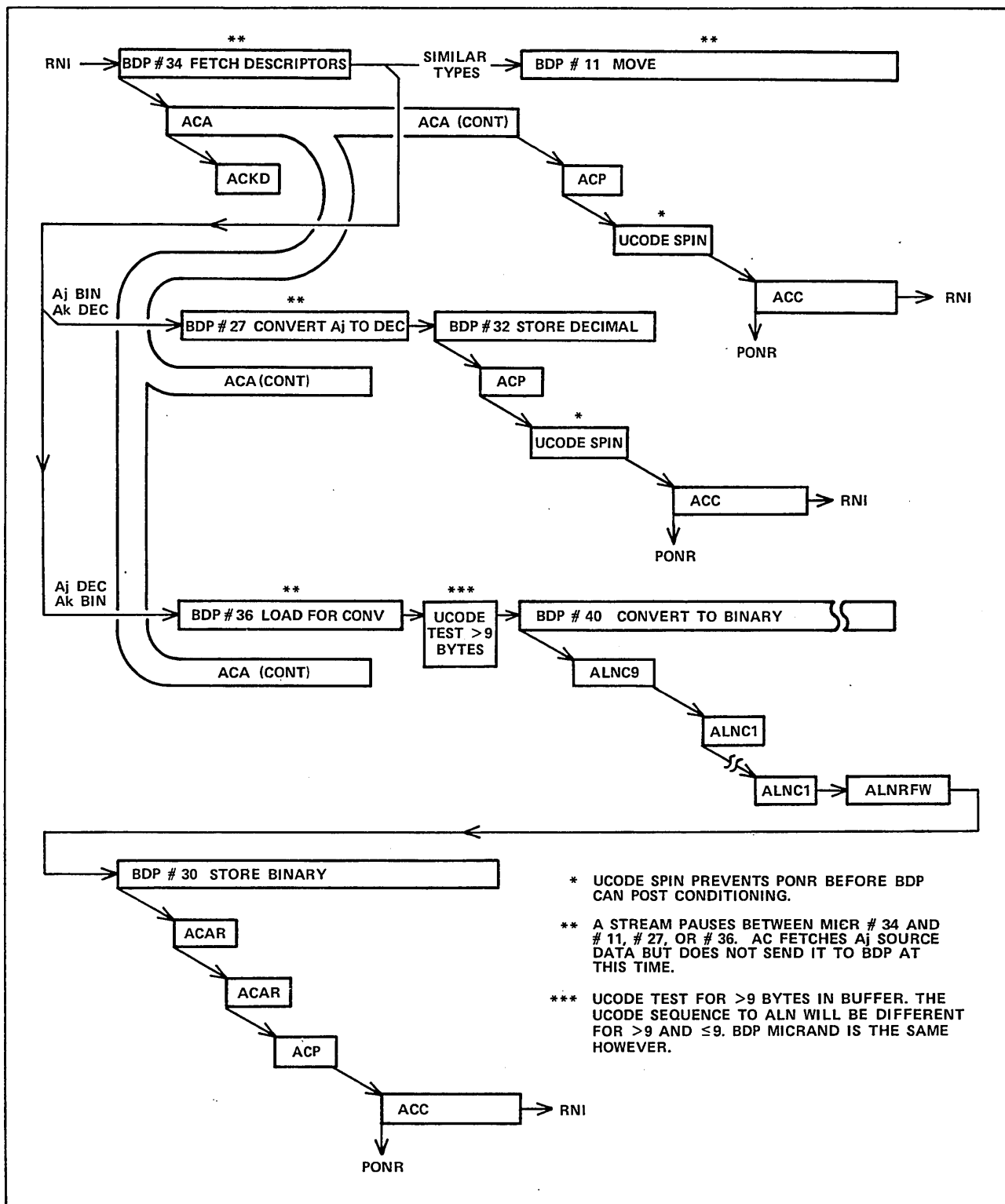


Figure 4-31. Numeric Move Micrands

This instruction moves decimal or binary data in the Aj source field to the Ak destination field after reformatting according to the Ak descriptor T field. The Aj and Ak data types determine which of three micrand sequences are executed.

- Standard Move - This sequence occurs if Aj and Ak are both binary or both decimal. Two BDP micrands control this sequence.
- Move After Convert Aj to Decimal - This sequence occurs if Aj is binary and Ak is decimal. Three BDP micrands control this sequence.
- Move After Convert Aj to Binary - This sequence occurs if Aj is decimal and Ak is binary. Four BDP micrands control this sequence.

#### First Micrand (#34) - All Moves

1. Pause Control sends Pause A Stream to AC.

#### NOTE

Because AC has no function that can load an Aj descriptor without also loading Aj data, the Aj data already loaded by AC must be kept in AC until BDP is issued a micrand that can process it. Hence, from the start of this micrand until the start of the next, BDP sends a constant Pause A Stream to AC. AC can load data but cannot disassemble data or transmit it to BDP.

2. AC loads Aj and Ak descriptors into BDP Aj, Ak Type, Length Registers.
3. BDP Condition Bit Select Mux sends BDP Condition (Aj and Ak data types decimal or binary) to CST. This allows CST to branch to appropriate microcode sequence.

#### Second Micrand (#11) - Standard Move

1. Pause Control drops Pause A Stream to AC.
2. AC A Stream Disassembly Mux sends Aj source decimal or binary byte to BDP A Stream Stages 1 through 3.
3. Resultant unpacked and unsigned decimal byte or sign-extended binary byte passes through A Stream Stage 4 and Decimal/Binary ALU.
4. Common Stages 5 and 6 reformat decimal data to destination data type. Binary data is not changed.
5. Common Stage 7 stores decimal or binary byte in Buffer RAM.
6. Steps 1 through 5 repeat until Aj source field is exhausted.

7. Buffer RAM sends decimal or binary byte to C Stream Stages 1 through 3.
8. If source was negative binary with an unsigned destination, C Stream Stages 1 and 2 perform two's complement on binary byte from Buffer RAM.
9. C Stream Stage 3 encodes decimal result to destination data type by forming ASCII bias, digit/byte sign, and slack digit.
10. C Stream Stages 4 and 5 encode translated decimal or binary byte (if necessary) and send Ak destination decimal or binary byte to AC C Stream Assembly Mux.
11. Steps 7 through 10 repeat until Buffer RAM is empty.

#### Second Micrand (#27) - Move After Convert Aj to Decimal

1. Pause Control drops Pause A Stream to AC.
2. AC A Stream Disassembly Mux sends Aj source binary byte to BDP A Stream Stages 1 through 3.
3. Sign-extended binary byte passes through A Stream Stage 4, Decimal/Binary ALU, and Common Stages 5 and 6.
4. Common Stage 7 stores binary byte in Buffer RAM.
5. Steps 1 through 4 repeat until Aj source field is exhausted.
6. Buffer RAM sends binary byte to C Stream Stages 1 through 3.
7. If source field is negative, C Stream Stages 1 and 2 perform one's complement on binary byte.

#### NOTE

The Binary/Decimal Converter expects binary data in its absolute form. Therefore, a negative source would require a two's complement at this point. Since the convert algorithm requires most significant data first, a two's complement cannot be performed because carry propagation is required. Instead, a one's complement is performed, leaving the absolute binary value one less than it ought to be. This error is corrected in the next micrand.

8. C Stream Stage 3 sends binary byte to Binary/Decimal Converter.
9. Binary/Decimal Converter performs binary to decimal conversion as described later under Convert to Decimal Sequence.
10. Steps 6 through 9 repeat until Buffer RAM is empty and entire decimal number is stored in Converter Result RAM.

Third Micrand (#32) - Move After Convert Aj to Decimal

1. Converter Result RAM sends Converted Decimal byte (least significant byte first) to A Stream Stages 1 through 3.

NOTE

Since the decimal result is packed decimal and the A Stream is expecting unpacked decimal, each decimal byte is sent twice: first with digits transposed and then normally. Transmission continues until all significant Converter Result RAM bytes are sent. The last digit is accompanied by last byte status.

2. Converted decimal byte passes through A Stream Stage 4.
3. If source field was negative, Decimal/Binary ALU adds one to result.
4. Common Stages 5 and 6 reformat decimal/byte to required destination field data type.
5. Common Stage 7 stores decimal byte in Buffer RAM.
6. Steps 1 through 5 repeat until the destination field length is satisfied.
7. Buffer RAM sends decimal byte to C Stream Stages 1 through 5.
8. C Stream Stage 3 encodes decimal result to destination data type by forming ASCII bias, digit/byte sign, and slack digit.
9. C Stream Stages 4 and 5 encode translated decimal byte (if necessary) and send Ak destination decimal byte to AC C Stream Assembly Mux.
10. Steps 7 through 9 repeat until Buffer RAM is empty.

Second Micrand (#36) - Move After Convert Aj to Binary

1. Pause Control drops Pause A Stream to AC.
2. AC A Stream Disassembly Mux sends Aj source decimal byte to BDP A Stream Stages 1 through 3.
3. Resultant unpacked and unsigned decimal byte passes through A Stream Stage 4 and Decimal/Binary ALU.
4. Common Stages 5 and 6 reformat byte to packed decimal.
5. Common Stage 7 stores packed decimal byte in Buffer RAM.
6. Steps 1 through 5 repeat until source field is exhausted.
7. BDP Condition Bit Select Mux sends BDP Condition (eight or less or more than nine significant bytes in Buffer RAM) to CST. This allows CST to issue appropriate microcode sequence for convert to binary.

Third Micrand (#40) - Move After Convert Aj to Binary

1. Buffer RAM sends packed decimal byte to C Stream Stages 1 through 3.
2. C Stream Stage 3 sends packed decimal byte to address Convert to Decimal/Binary RAM.
3. Convert to Decimal/Binary RAM sends BDP Converted Binary byte to ALN.
4. ALN completes decimal to binary conversion as described later under Convert to Binary Sequence.
5. Steps 1 through 4 repeat until Buffer RAM is empty.
6. ALN sends binary equivalent of decimal data to Register File.

Fourth Micrand (#30) - Move After Convert Aj to Binary

1. Register File sends 64-bit unsigned binary word to AC A Stream Disassembly network.
2. AC A Stream Disassembly Mux sends unsigned binary byte to BDP A Stream Stages 1 through 3.
3. Unsigned binary byte passes through A Stream Stage 4, Decimal/Binary ALU, and Common Stages 5 and 6.
4. Common Stage 7 stores unsigned binary byte in Buffer RAM.
5. Steps 1 through 4 repeat until entire unsigned binary field is stored in Buffer RAM.

NOTE

Register File sends two 64-bit words to AC.  
First, for least significant word. Second,  
for most significant word.

6. Buffer RAM sends binary byte to C Stream Stages 1 through 5.
7. If source field is negative, C Stream Stages 1 and 2 perform two's complement on binary byte.
8. C Stream Stages 4 and 5 encode translated binary byte (if necessary) and send Ak destination binary byte to AC C Stream Assembly Mux.
9. Steps 6 through 8 repeat until Buffer RAM is empty.



#### Decimal Scale (E4jkID)

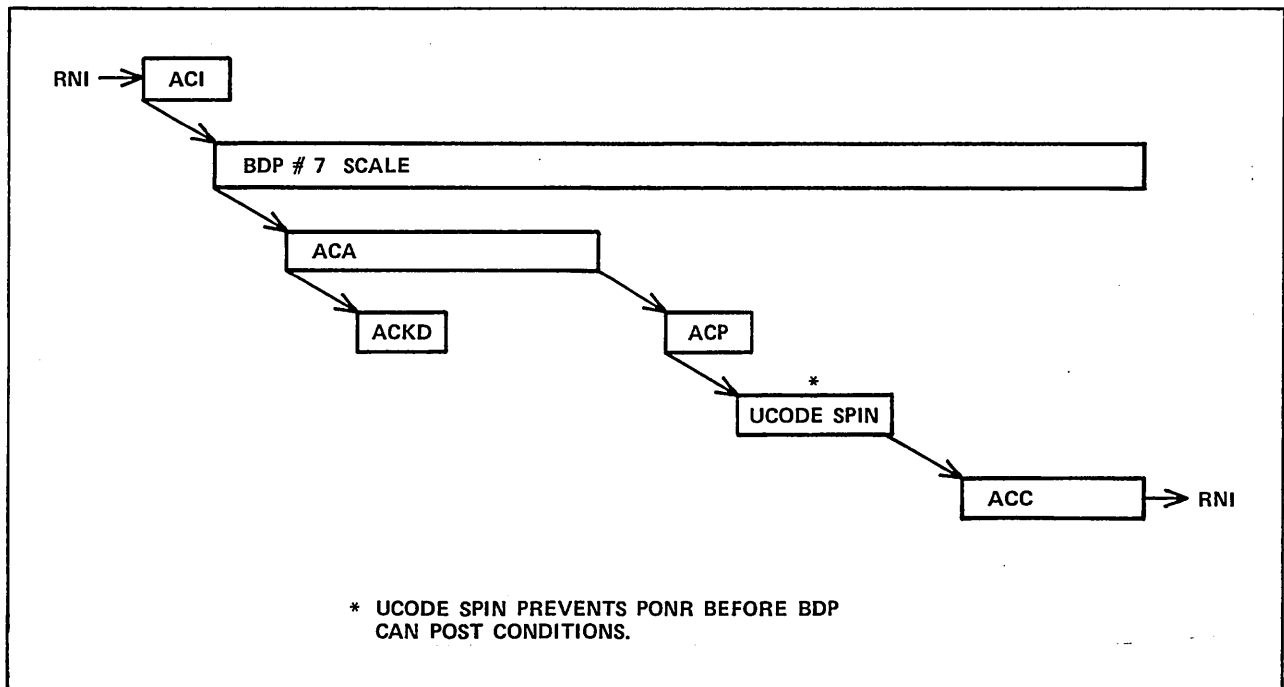


Figure 4-32. Decimal Scale Micrands

This instruction moves decimal data in the Aj source field to the Ak destination field after shifting according to an 8-bit scale (shift count). AC forms the shift count by adding Xi Register bits 56 through 63 to the sign-extended D instruction field. Positive shift counts (bit 56 clear) cause left shifts according to the value of bits 57 through 63 (0 through 127 places). Negative shift counts (bit 56 set) cause right shifts according to two's complement value of bits 57 through 63 (0 through 128 places). Both shifts are end-off with zero-fill. The shift is a digit shift relating to the decimal value of the source, and not relating to sign bytes, slack digits, and packed or unpacked data. For example, a left shift of one multiplies the source absolute value by 10, a shift of two by 100, and so on. A right shift divides by 10, 100, and so on. One BDP micrand causes the following data flow.

1. AC B Stream Disassembly Mux sends shift count to BDP Scale Control.
2. AC A Stream Disassembly Mux sends Aj source data byte to BDP A Stream Stages 1 through 3.
3. A Stream Stage 4 sends resultant unpacked and unsigned decimal byte to Decimal/Binary ALU.
4. Decimal/Binary ALU shifts decimal byte left, right, or not at all according to Shift Left, Right from Scale Control.

#### NOTE

The shift is controlled by a shift counter in Scale Control. If a left shift, the counter decrements to zero; if a right shift, it increments to zero. In either case, the shift is complete when the shift counter reaches zero.

On a left shift, BDP immediately pauses the AC A stream in order to pad zero digits on the right. A fake BDP A Stream Stage 3 active signal is generated for each shift count which streams that number of zero digits toward the Buffer RAM. When the shift count is satisfied, Pause A Stream drops and A<sub>j</sub> source digits stream into the Buffer RAM. If the shift count was originally zero, Pause A Stream would not have activated and data would have streamed normally.

On a right shift, the shift count increments each time a source digit appears in A Stream Stage 3. At the same time, the stage 3 active status is blocked from passing into stage 4. Therefore, these digits are lost. When the shift count is satisfied, data streams normally toward the Buffer RAM.

5. Common Stages 5 and 6 reformat decimal byte to appropriate destination field data type.
6. Common Stage 7 stores decimal byte in Buffer RAM.
7. Steps 1 through 6 repeat until both source fields are exhausted.
8. Buffer RAM sends data byte to C Stream Stages 1 through 5.
9. C Stream Stage 3 encodes decimal result to destination data type by forming ASCII bias, digit/byte sign, and slack digit.
10. C Stream Stages 4 and 5 encode translated decimal byte (if necessary) and send A<sub>k</sub> destination decimal byte to AC C Stream Assembly Mux.
11. Steps 8 through 10 repeat until Buffer RAM is empty.

Decimal Scale Rounded (E5jkiD)

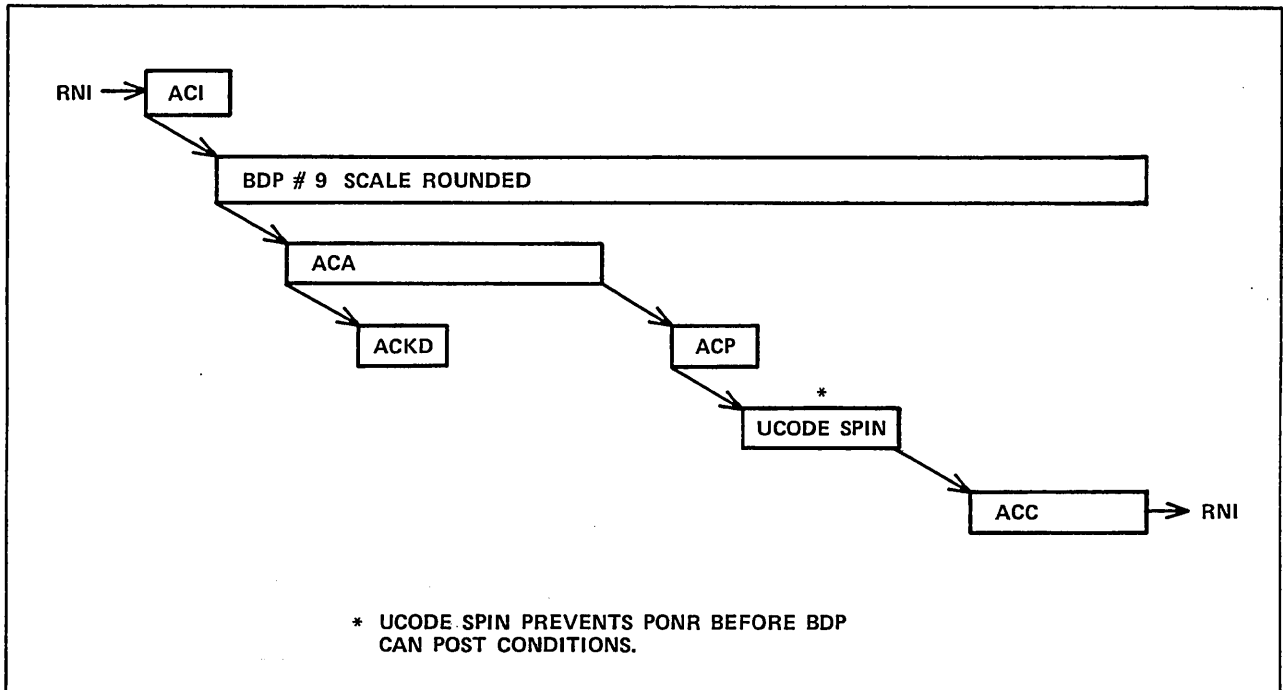


Figure 4-33. Decimal Scale Rounded Micrands

This instruction is the same as the decimal scale (E4jkiD) instruction except on right shifts. Rounding is accomplished on right shifts by remembering the value of the last digit shifted end-off. If five or greater, one is added to the remaining source absolute value. The Decimal/Binary ALU is always in decimal add mode with Aj source data added to zero B Stream data. If a round is called for, a carry enters the ALU on the rightmost digit (after shifting). If necessary, the carry propagates from digit to digit.

## BYTE INSTRUCTION SEQUENCES

### Move Bytes (76.jk)

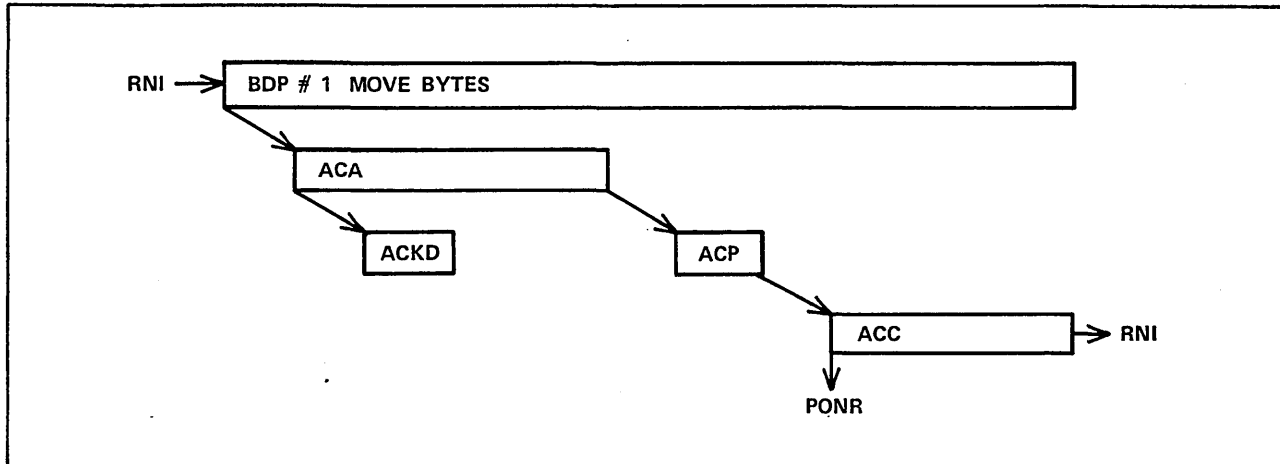


Figure 4-34. Move Bytes Micrands

This instruction moves data of any type in the Aj source field to the Ak destination field. The data is not modified. One BDP micrand causes the following data flow.

1. AC A Stream Disassembly Mux sends Aj source data byte to BDP A Stream Stages 1 through 3.
2. Byte passes through A Stream Stage 4, Decimal/Binary ALU, and Common Stages 5 and 6.
3. Common Stage 7 stores byte in Buffer RAM.
4. Steps 1 through 3 repeat until Buffer RAM contains enough source or fill (20 hexadecimal) bytes to satisfy the Ak destination length.
5. Buffer RAM sends byte through C Stream Stages 1 through 5 to AC C Stream Assembly Mux.
6. Step 5 repeats until Buffer RAM is empty.

Byte Compare (77jk)

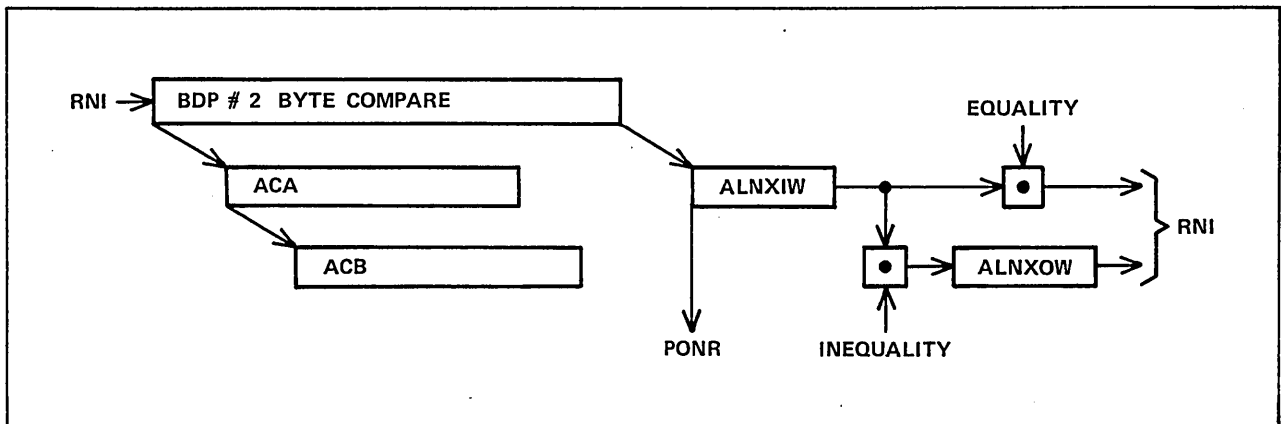


Figure 4-35. Byte Compare Micrand

This instruction compares byte strings of any legal type in the Ak and Aj source fields. The first unequal byte pair ends the instruction and causes status to be sent to the XO and XI Registers. One BDP micrand causes the following data flow.

1. AC A Stream Disassembly Mux sends Aj source data byte to BDP A Stream Stages 1 through 3.
2. AC B Stream Disassembly Mux sends Ak source data byte to BDP B Stream Stages 1 through 3.
3. A and B Stream Stage 4 synchronize bytes.
4. Decimal/Binary ALU compares source bytes from A and B Stream Stage 4 by performing two's complement subtract.
5. Common Stage 5 sends comparison signals to Register File Control and difference to Common Stage 6.

NOTE

Difference enters Buffer RAM through Common Stage 7 but is not used. However, Buffer RAM Address indicates the sequence number of the unequal bytes.

6. Steps 1 through 5 repeat until a miscompare occurs or both sources are exhausted. Shorter field is right-filled with spaces (20 hexadecimal).
7. X0 Result Selector sends Buffer RAM Address (sequence count) to OPI.
8. X1 Result Selector sends BDP X1 Result to OPI as follows.

Compare	X1 Result			
	32	33	34	63
A = B	0	0	0	---
A > B	0	1	0	---
A < B	1	1	0	---

9. BDP Condition Bit Select Mux sends BDP Condition (A equal or not equal to B) to CST.
10. If A and B are not equal, CST enables BDP X0 Result to X0 Register. If equal, X0 Register remains unchanged.

#### Byte Compare Collated (E9jkID)

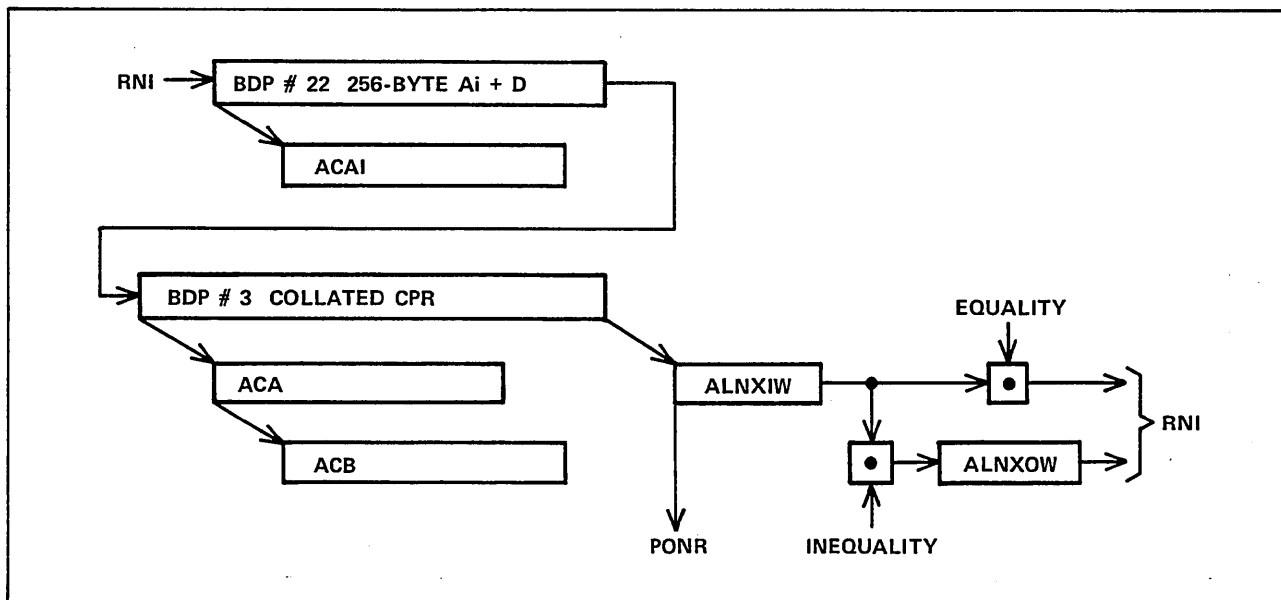


Figure 4-36. Byte Compare Collated Micrandis

This instruction compares byte strings of any legal type in the Ak and Aj source fields after translating according to an Ai plus D translation table. The first unequal translated byte pair ends the instruction and causes status to be sent to the X0 and X1 Registers. Two BDP micrandis cause the following data flow.

First Micrand (#22)

1. AC A Stream Disassembly Mux sends Ai plus D translation table byte to BDP A Stream Stages 1 through 3.
2. Register File A Stream Stage 4 stores translation table byte in address 00 of Register File A and B.
3. Steps 1 and 2 repeat until Register File A Address Counter increments to FF (hexadecimal). This indicates that 256 translation table bytes have been stored in both files.

Second Micrand (#3)

1. AC A Stream Disassembly Mux sends Aj source data byte to BDP A Stream Stages 1 through 3.
2. AC B Stream Disassembly Mux sends Ak source data byte to BDP B Stream Stages 1 through 3.
3. Register File A and B Stream Stage 4 synchronize source bytes and address Register File A and B RAMs.
4. Register File Compare Control compares translation table bytes at Register File A and B RAM addresses.
5. If translation table bytes are equal, steps 1 through 4 repeat until a miscompare occurs or both sources are exhausted. Shorter field is right-filled with spaces (20 hexadecimal).
6. XO Result Selector sends Buffer RAM Address to OPI.

NOTE

Buffer RAM is not used for this instruction. However, Buffer RAM Address Counter indicates the sequence number of the unequal translation table bytes.

7. X1 Result Selector sends BDP X1 Result to OPI as follows.

	<u>X1 Result</u>			
<u>Compare</u>	<u>32</u>	<u>33</u>	<u>34</u>	<u>63</u>
A = B	0	0	0	--- 0
A > B	0	1	0	--- 0
A < B	1	1	0	--- 0

8. BDP Condition Bit Select Mux sends BDP Condition (A equal or not equal to B) to CST.
9. If A and B are not equal, CST enables BDP XO Result to XO Register. If equal, XO Register remains unchanged.

### Byte Translate (EBjkID)

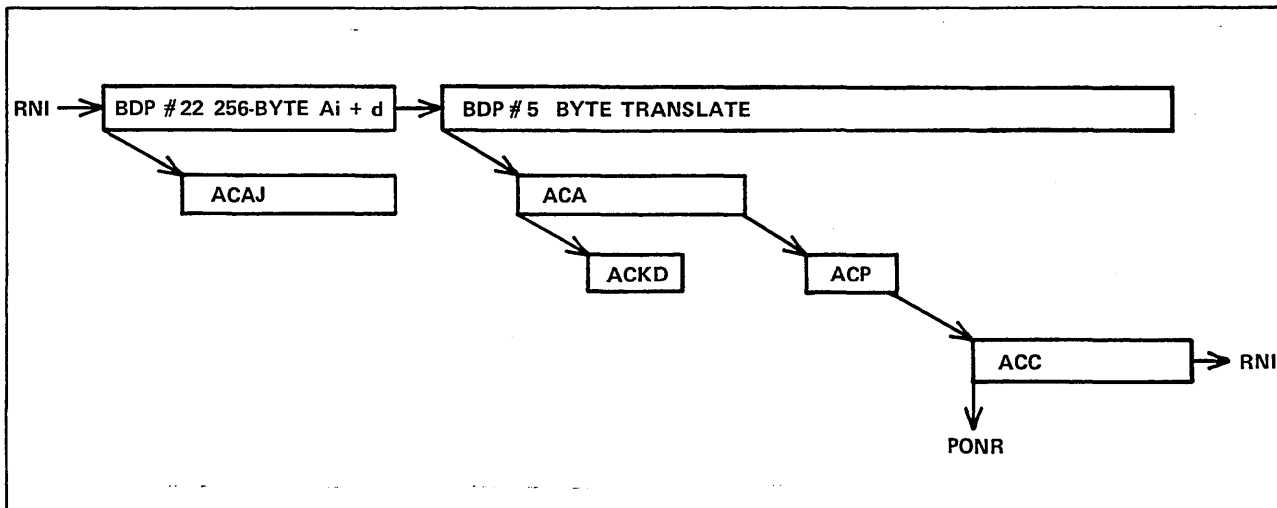


Figure 4-37. Byte Translate Micrand

This instruction moves data of any legal type in the  $A_j$  source field to the  $A_k$  destination field after translating according to an  $A_i$  plus  $D$  translation table. Two BDP micrands cause the following data flow.

#### First Micrand (#22)

1. AC A Stream Disassembly Mux sends  $A_i$  plus  $D$  translation table byte to BDP A Stream Stages 1 through 3.
2. Register File A Stream Stage 4 stores translation table byte in address 00 of Register Files A and B.

#### NOTE

Translation table is stored in Register File B but not used.

3. Steps 1 and 2 repeat until Register File A Address Counter increments to FF (hexadecimal). This indicates that 256 translation table bytes have been stored in both files.



Second Micrand (#5)

1. AC A Stream Disassembly Mux sends Aj source data byte to BDP A Stream Stages 1 through 3.
2. Byte passes through Register File A Stream Stage 4 and addresses Register File A RAM.
3. Register File A RAM sends translation table byte to Common Stage 7.
4. Common Stage 7 stores translation table byte in Buffer RAM.
5. Steps 1 through 4 repeat until Buffer RAM contains enough translation table bytes (including translated blanks) to satisfy the Ak destination length.
6. Buffer RAM sends translation table byte through C Stream Stages 1 through 5 to AC C Stream Assembly Mux.
7. Step 6 repeats until Buffer RAM is empty.

Edit (EDjkiD)

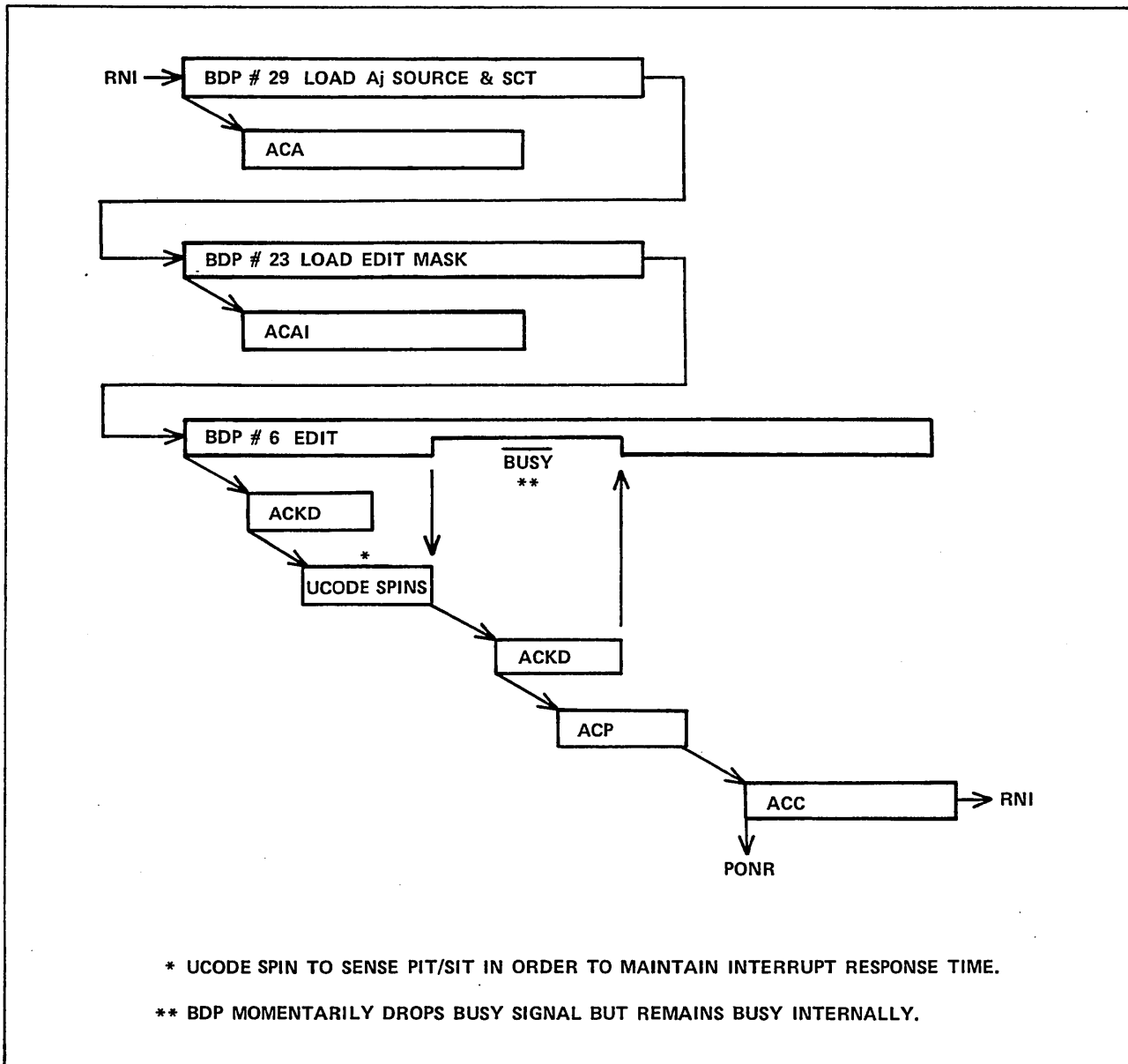


Figure 4-38. Edit Micrands

This instruction edits numeric or alphanumeric data in the Aj source field according to the Ai plus D edit mask and places the result in the Ak destination field. Three BDP micrands cause the following data flow.

First Micrand (#29)

1. AC A Stream Disassembly Mux sends Aj source decimal or alphanumeric byte to BDP A Stream Stages 1 through 3.
2. Register File A Stream Stage 4 stores resultant unpacked and unsigned decimal byte or unchanged alphanumeric byte in address 00 of Register Files A and B.

NOTE

Source data is stored in Register File A but will be rewritten by edit mask during second micrand.

3. Steps 1 and 2 repeat until source field is exhausted.
4. Simultaneously with steps 1 through 3, Edit SCT RAM preloads with SCT values. Micrand is complete when source field is exhausted and SCT RAM is preloaded.

NOTE

The entire source field is read before editing begins for two reasons. First, some micro-operations (MOPs) require the source field sign be available before editing begins. If source field is signed, the sign is latched and retained for use during the third micrand. Second, any invalid decimal data must be reported, even if the third micrand determines that the edit is a no operation.

The number of source field bytes stored is the number of alphanumeric bytes or the number of decimal digits contained in the source plus one or more extra fill bytes. Register File B Bit 9 represents source exhausted status to assist the third micrand in determining end of source. This bit will be set in the one or more fill bytes stored after the source. If an alphanumeric source has a length of 256 and, therefore, no room in Register File B for a fill byte, the third micrand determines end of source by noting that Register File B Address Counter attempted to increment past 256.

#### Second Micrand (#23)

1. AC A Stream Disassembly Mux sends Ai plus D edit mask byte to BDP A Stream Stages 1 through 3.
2. Register File A Stream Stage 4 stores edit mask byte in address 00 of Register File A.
3. Steps 1 and 2 repeat until Register File A Address Counter increments to value of first edit mask byte.

#### NOTE

The first edit mask byte contains the mask length. It is latched up after passing through stage 4. This byte is used to terminate this micrand and is also used in the third micrand. AC also uses this byte to control the number of bytes sent to BDP.

#### Third Micrand (#6)

1. Register File A RAM sends edit mask byte to Edit Control.

#### NOTE

This first edit mask byte indicates mask length and is not sent to Edit Control.

2. Edit Control interprets MOP/specification value (SV) in edit mask byte and takes appropriate action.
3. Edit Result Mux forms destination data by using or manipulating source data, edit mask data, special character table (SCT) data, or symbol mask (SM) data.
4. Common Stage 7 stores Edit Result in Buffer RAM.

#### NOTE

MOP F clears the Buffer RAM Address Counter. This allows rewriting previously written data.

5. Steps 1 through 4 repeat until edit mask is exhausted or under control of MOP F.

#### NOTE

The number of destination bytes created by Edit may be less than specified by the Ak descriptor length. If so, only those bytes created by Edit are allowed to be written in CM. Therefore, both AC and BDP Ak length circuits are set to the shorter value before the C Stream operation begins.

To facilitate this, the BDP Busy signal to ICP is momentarily dropped allowing the spinning micrand to manipulate the new length. The Buffer RAM Address Counter is equal to the number of bytes actually edited and is constantly transmitted on the BDP X1 Result lines. Microcode uses this value and generates a new descriptor. It then initiates an AC B Stream descriptor-only operation, thereby changing the Ak length in both AC and BDP. BDP goes back to busy and begins the C stream output of the Buffer RAM.

#### Byte Scan While Nonmember (F3jkID)

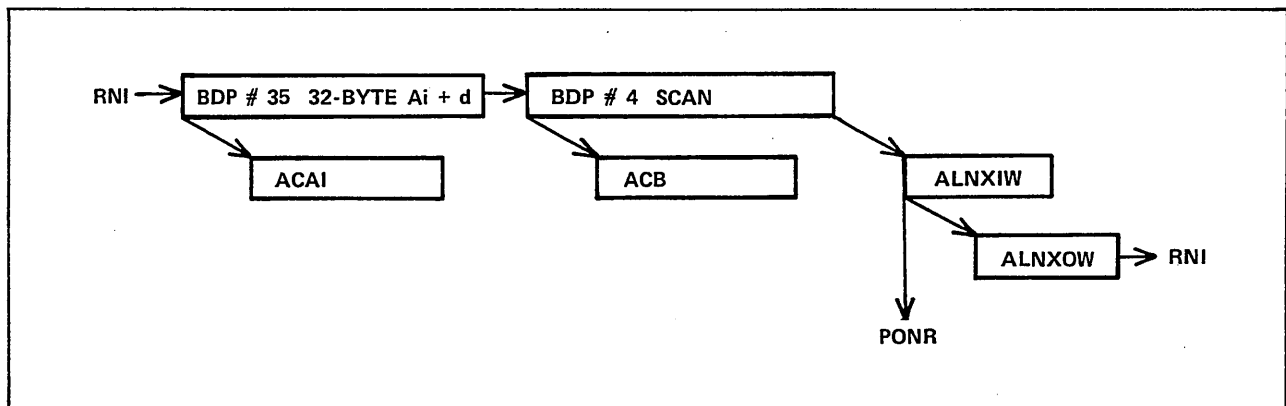


Figure 4-39. Byte Scan While Nonmember Micrands

This instruction inspects character (byte) strings of any legal type in the Ak source field for specific byte values using an Ai plus D scan table. The first specific byte value encountered ends the instruction and causes status to be sent to the X0 and X1 Registers. Two BDP micrands cause the following data flow.

#### First Micrand (#35)

1. AC A Stream Disassembly Mux sends Ai plus D scan table byte to BDP A Stream Stages 1 through 3.
2. Register File A Stream Stage 4 stores scan table byte in address 00 of Register Files A and B.

#### NOTE

Scan table is stored in Register File A but not used.

3. Steps 1 and 2 repeat until Register File A Address Counter increments to 1F (hexadecimal). This indicates that 32 scan table bytes have been stored in both files.

#### Second Micrand (#4)

1. AC B Stream Disassembly Mux sends Ak source data byte to BDP B Stream Stages 1 through 3.
2. Register File A Stage 4 performs right shift three places and inserts zeros on left of byte which addresses Register File B RAM.

#### NOTE

This allows addressing of 32 bytes in Register File B RAM. The three bits shifted end-off later select one of eight bits from the addressed byte.

Unchanged source data byte also passes through B Stream Stage 4, Decimal/Binary ALU, and Common Stage 5.

3. Register File B RAM sends addressed byte to Scan Hit Control. Common Stage 6 sends Bits 5 through 7 (rightmost three bits) of source data byte to Scan Hit Control.
4. Scan Hit Control selects one of eight Register File B Data bits based on value of Common Stage 6 Bits 5 through 7.
5. Scan Hit Control determines that selected Register File B Data bit is set (Scan Hit) or clear (no Scan Hit).

If Scan Hit occurs on this source byte, instruction ends when X0 Result Selector sends Buffer RAM Address and X1 Result Selector sends Common Stage 7 Data to OPI.

# NOTE

Buffer RAM is not used for this instruction. However, Buffer RAM Address Counter indicates the sequence number of the hit byte.

If no Scan Hit occurs on this source byte, steps 1 through 5 repeat until Scan Hit occurs or source field is exhausted. If exhausted, instruction ends when X0 Result Selector sends Ak Descriptor Length Field and X1 Result Selector sends following no hit status to OPI.

32	33	34	63
1	0	0 - - -	0

## CALCULATE SUBSCRIPT (F4jkid) INSTRUCTION SEQUENCE

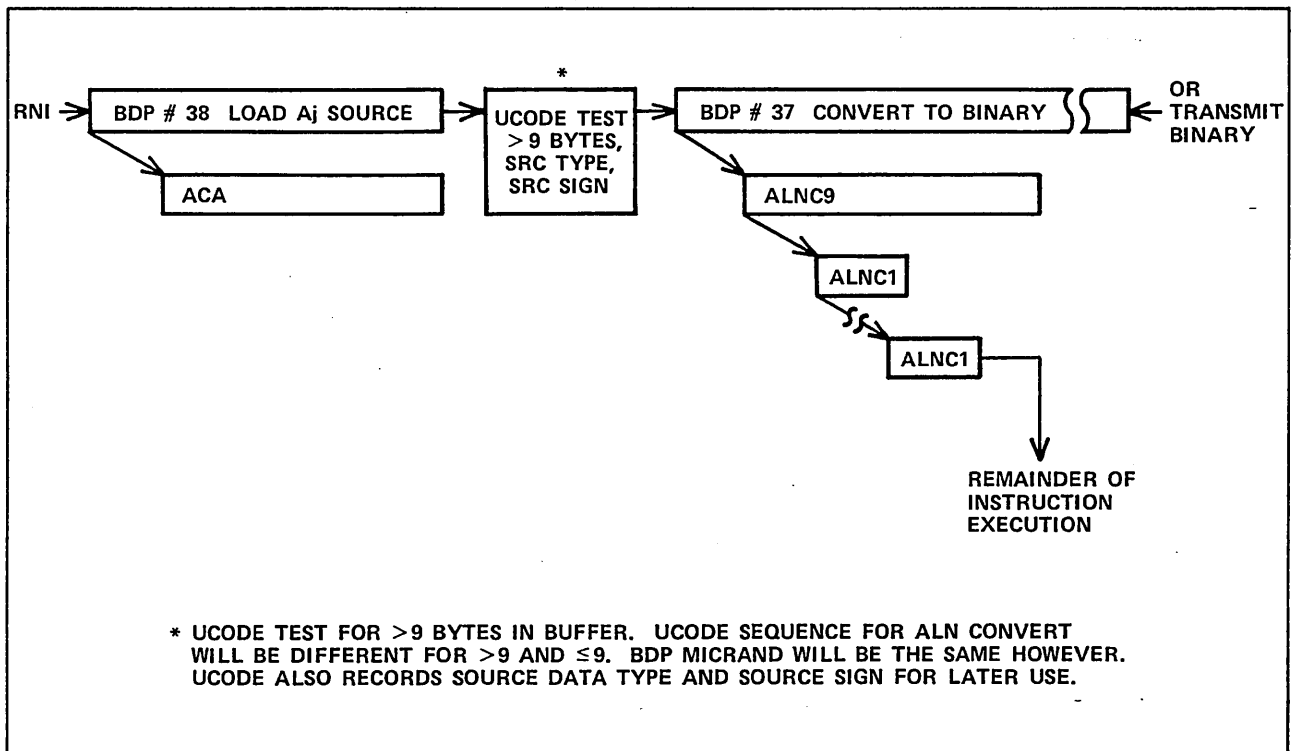


Figure 4-40. Calculate Subscript Micrands

BDP's participation in this instruction is limited to reading a decimal or binary byte string and transmitting it to ALN which forms an integer. Two BDP micrands cause the following data flow.

#### First Micrand (#38)

1. AC A Stream Disassembly Mux sends Aj source decimal or binary byte to BDP A Stream Stages 1 through 3.
2. Resultant unpacked and unsigned decimal byte or unchanged binary byte passes through A Stream Stage 4 and Decimal/Binary ALU.
3. Common Stages 5 and 6 reformat decimal byte to packed unsigned decimal.
4. Common Stage 7 stores packed decimal or binary byte in Buffer RAM.
5. Steps 1 through 4 repeat until Aj source field is exhausted.
6. Address of most significant nonzero byte enters Buffer RAM Address Counter.
7. BDP Condition Bit Select Mux sends BDP Condition (source field decimal or binary, source field negative or positive, eight or less or more than nine significant bytes in Buffer RAM) to CST. This allows CST to issue appropriate ALN micrand(s) to accompany next BDP micrand.

#### Second Micrand (#37)

1. Buffer RAM sends packed decimal or binary byte to C Stream Stages 1 through 3.
2. If source field is negative binary, C Stream Stages 1 and 2 complement byte.

#### NOTE

If the source is binary and negative, it must undergo a two's complement since ALN expects the absolute value. Since data is read from Buffer RAM most significant byte first, a two's complement cannot be performed because this requires a one's complement with a forced carry in. Instead, a one's complement without the forced carry is performed. In this case, the resulting integer in ALN is one less than the original source value. CST senses the Aj source sign and adds one to the integer if the source is negative.

3. If packed decimal, C Stream Stage 3 sends byte to address Convert to Decimal/Binary RAM.  
  
If binary, C Stream Stage 3 Data bypasses Decimal/Binary RAM and goes to ALN as BDP Converted Binary byte.
4. If packed decimal, Convert to Decimal/Binary RAM sends BDP Converted Binary byte to ALN.



5. If packed decimal, ALN completes decimal-to-binary conversion as described later under Convert to Binary Sequence.

If binary, ALN assembles bytes into an integer of absolute value by adding each byte to a cumulative result.

6. Steps 1 through 5 repeat until Buffer RAM is empty.

#### IMMEDIATE DATA INSTRUCTION SEQUENCES

##### Move Immediate Data (F9jkiD)

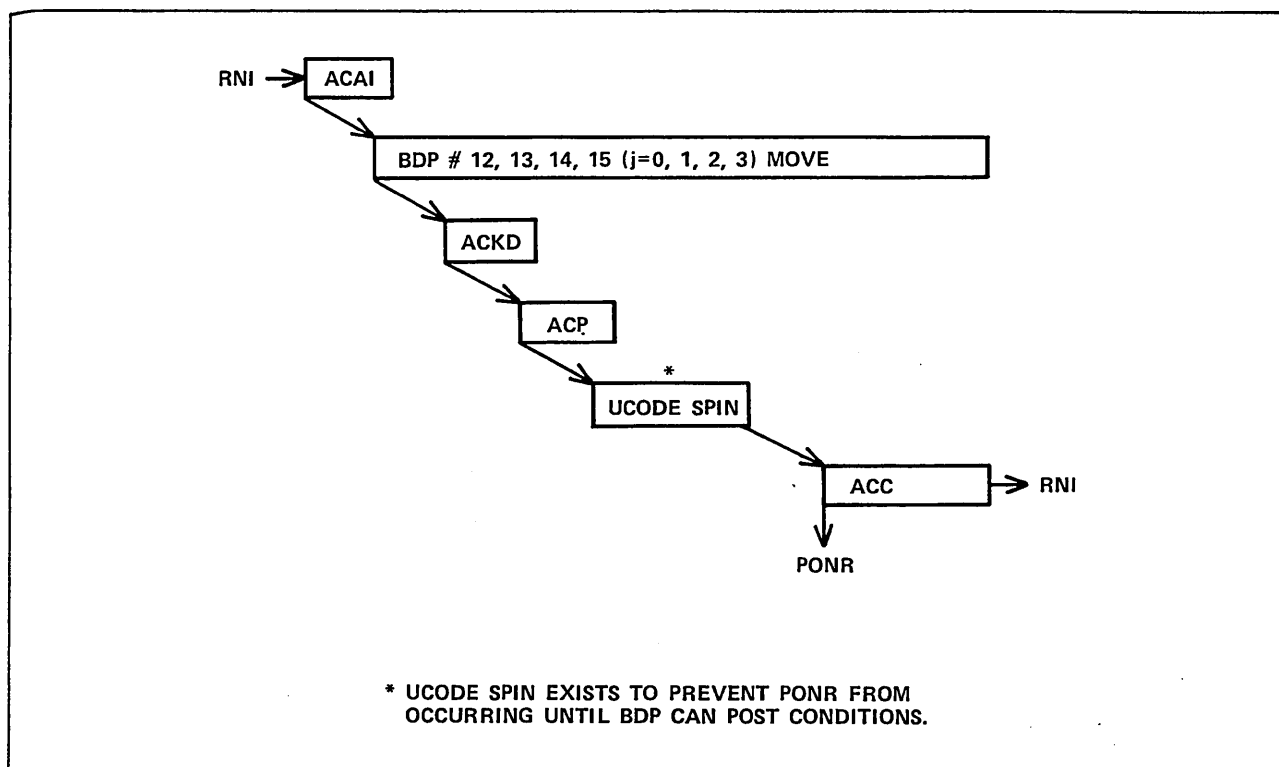


Figure 4-41. Move Immediate Data Micrands

This instruction moves the Xi plus D immediate data byte to the Ak destination field according to the j field as follows.

<u>j Field</u>	<u>Operation</u>
0	Move binary immediate byte followed by zero bytes to destination field (right to left).
1	Move decimal immediate byte followed by zero bytes to destination field (right to left).

<u>j Field</u>	<u>Operation</u>
2	Move alphanumeric character in immediate byte to each byte in destination field (left to right).
3	Move alphanumeric character in immediate byte followed by blank characters to destination field (left to right).

One BDP micrand causes the following data flow.

1. AC B Stream Disassembly Mux sends immediate byte to BDP Scale Control.

#### NOTE

Counter in Scale Control serves as holding register for immediate byte during immediate data instructions.

2. Scale Control sends immediate byte to A Stream Stages 1 through 3.
3. Immediate byte passes through A Stream Stage 4 and Decimal/Binary ALU.
4. Common Stages 5 and 6 reformat decimal or binary data to destination data type. Alphanumeric data is not changed.
5. Common Stage 7 stores immediate byte in Buffer RAM.
6. A Stream Stage 3 sends zero byte (j equals 0 or 1), immediate byte (j equals 2), or blank character (j equals 3) to A Stream Stage 4.
7. Zero byte, immediate byte, or blank character passes through Decimal/Binary ALU, Common Stages 5 through 7, and enters Buffer RAM.
8. Steps 6 and 7 repeat until destination length is satisfied.
9. Buffer RAM sends immediate byte to C Stream Stages 1 through 3.
10. C Stream Stages 4 and 5 encode translated decimal or binary byte (if necessary) and send Ak destination decimal, binary, or alphanumeric byte to AC C Stream Assembly Mux.
11. Steps 9 and 10 repeat for zero byte, immediate byte, or blank character until Buffer RAM is empty.

#### Compare Immediate Data (FAjkiD)

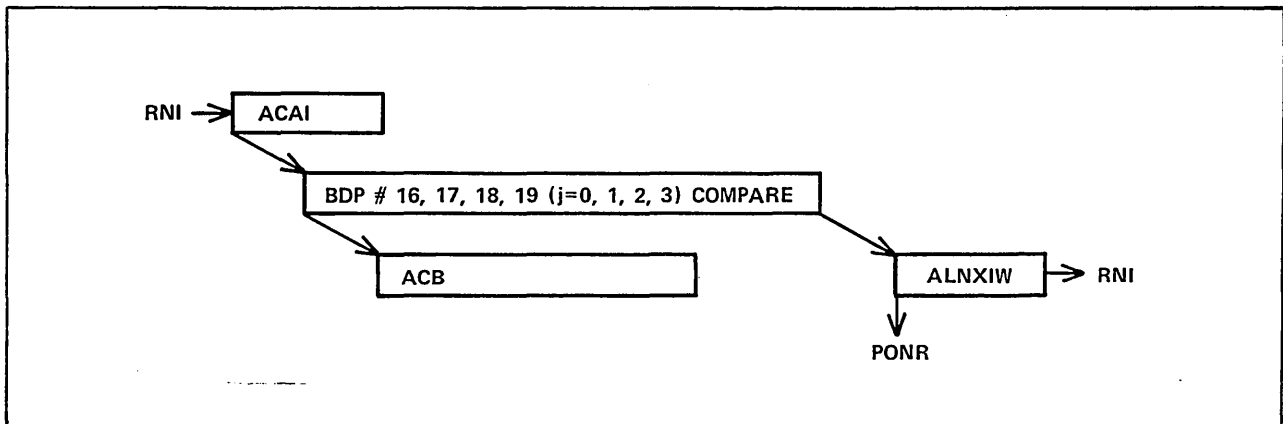


Figure 4-42. Compare Immediate Data Micrands

This instruction compares the Xi plus D immediate data byte with the Ak source field according to the j field as follows and reports status to the X1 Register.

<u>j Field</u>	<u>Operation</u>
0	Compare binary immediate byte followed by zero bytes with entire Ak source field (right to left).
1	Compare decimal immediate byte followed by zero bytes with entire Ak source field (right to left).
2	Compare alphanumeric character in immediate byte with each byte in Ak source field (left to right).
3	Compare alphanumeric character in immediate byte followed by blank bytes with corresponding bytes in Ak source field (left to right).

One BDP micrand causes the following data flow.

1. AC B Stream Disassembly Mux sends immediate byte to BDP Scale Control.

#### NOTE

Counter in Scale Control serves as holding register for immediate byte during immediate data instructions.

2. Scale Control sends immediate byte to A Stream Stages 1 through 3.
3. AC B Stream Disassembly Mux sends Ak source binary, decimal, or alphanumeric byte to B Stream Stages 1 through 3.
4. A and B Stream Stage 4 synchronizes binary, decimal, or alphanumeric bytes.

5. Decimal/Binary ALU compares binary, decimal, or alphanumeric bytes from A and B Stream Stage 4.
6. Common Stage 5 sends comparison signals (Carry Out, Equals Zero) to Register File Compare Control.
7. If j equals 0 or 1, Decimal/Binary ALU compares zero byte from A stream with next binary (0) or decimal (1) byte from B stream. This repeats from right to left until Ak source field is exhausted.

If j equals 2 or 3, Decimal/Binary ALU compares immediate byte (2) or blank byte (3) from A stream with next alphanumeric byte from B stream. This repeats from left to right until miscompare occurs or Ak source field is exhausted.

8. Register File Compare Control determines result of comparison.
9. Xl Result Selector sends BDP Xl Result to OPI as follows.

	<u>Xl Result</u>			
<u>Compare</u>	32	33	34	63
A = B	0	0	0	---
A > B	0	1	0	---
A < B	1	1	0	---

#### Add Immediate Data (FBjkiD)

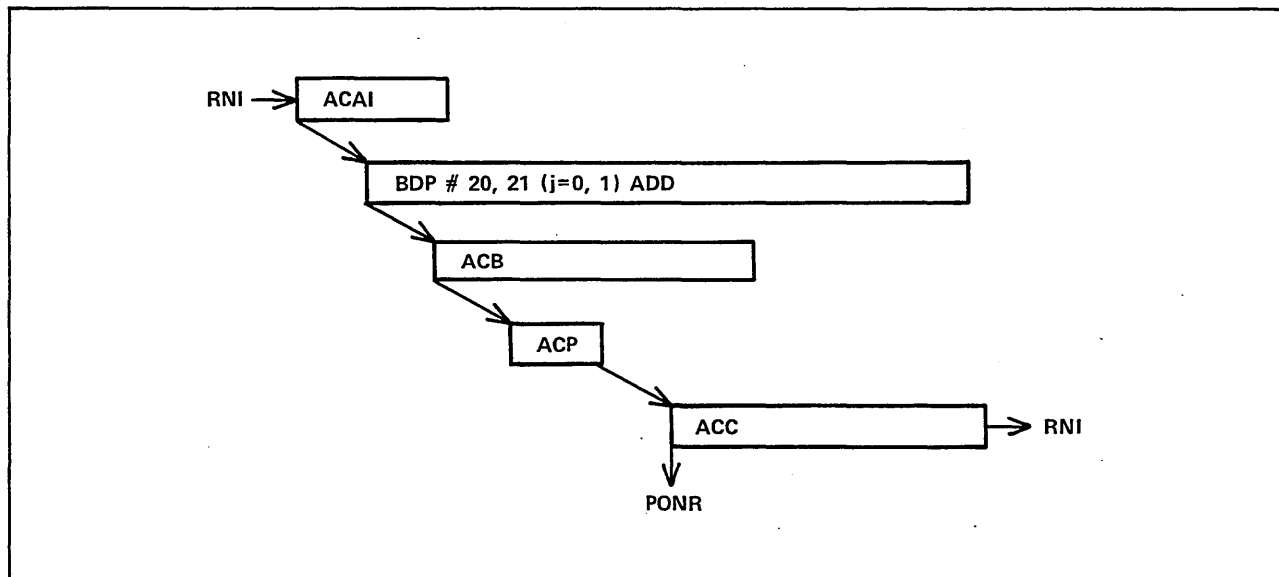


Figure 4-43. Add Immediate Data Micrands

This instruction adds the Xi plus D immediate data byte to the Ak source field according to the j field and places the sum in the Ak destination field.

<u>j Field</u>	<u>Operation</u>
0	Add binary immediate byte followed by zero bytes to Ak source field (right to left).
1	Add decimal immediate byte followed by zero bytes to Ak source field (right to left).

One BDP micrand causes the following data flow.

1. AC B Stream Disassembly Mux sends immediate byte to BDP Scale Control.

NOTE

Counter in Scale Control serves as holding register for immediate byte during immediate data instructions.

2. Scale Control sends immediate byte to A Stream Stages 1 through 3.
3. AC B Stream Disassembly Mux sends Ak source binary or decimal byte to B Stream Stages 1 through 3.
4. A and B Stream Stage 4 synchronize binary or decimal bytes.
5. Decimal/Binary ALU adds binary or decimal bytes.
6. Common Stages 5 and 6 reformat binary or decimal sum byte to appropriate destination field data type.
7. Common Stage 7 stores binary or decimal byte in Buffer RAM.
8. A Stream Stage 3 sends zero byte to stage 4 and steps 3 through 7 repeat until Ak source field is exhausted.
9. Buffer RAM sends binary or decimal byte to C Stream Stages 1 through 5.
10. C Stream Stage 3 encodes decimal result to destination data type by forming ASCII bias, digit/byte sign, and slack digit.
11. C Stream Stages 4 and 5 encode translated binary or decimal byte (if necessary) and send Ak destination binary or decimal byte to AC C Stream Assembly Mux.
12. Steps 9 through 11 repeat until Buffer RAM is empty.

#### CONVERT-TO-BINARY SEQUENCE

BDP and ALN share the task of converting decimal data to binary data. BDP shown in BDP 1.0 converts a string of packed decimal bytes stored in the Buffer RAM to an equivalent string of binary bytes using the Convert to Binary/Decimal RAM. ALN adds each binary byte from BDP to a running total and multiplies by the binary equivalent of 100 (decimal) to produce a binary result of one or two 64-bit words. ALN stores the result in one or two Register File locations in OPI for later use.

Common Stage 7 originally stores packed decimal bytes (regardless of source decimal type) in the Buffer RAM, from right to left (least significant byte first), starting at address zero. After the bytes are stored, the Buffer RAM Address Counter sets to the address of the most significant nonzero byte. This eliminates any leading zeros from the conversion. The counter decrements from this address to zero when the last byte is in the Buffer RAM. Also after the bytes are stored, the BDP Condition Bit Select Mux informs CST that one through nine or more than nine packed decimal bytes are to be converted. Nine packed decimal bytes is the largest decimal number that fits in a single 64-bit binary word. ALN converts one through nine packed decimal bytes using a single micrand. However, ALN requires another micrand for each additional byte to be converted. Regardless of the number of bytes, BDP uses only one micrand to cause the following data flow.

1. Buffer RAM sends packed decimal byte (most significant first) to address Convert to Binary/Decimal RAM.
2. Convert to Binary/Decimal RAM sends BDP Converted Binary byte (representing two packed decimal digits) to ALN.
3. ALN adds BDP Converted Binary byte to running binary total (initially zero).
4. ALN multiplies running binary total by 1100100 (100 decimal).
5. Steps 1 through 4 repeat until last or ninth byte (whichever is first) is added to running binary total. Last byte is added but running total is not multiplied unless more than nine bytes are involved.
6. ALN sends first 64-bit binary word to Register File in OPI.
7. Steps 1 through 4 repeat for nine or more bytes until last byte is added to running binary total. Last byte is added but running total is not multiplied.

#### NOTE

For more than nine bytes, BDP continues using the same micrand while CST and ALN go into a stepping sequence (one micrand for each byte converted). CST samples BDP XO Result which contains a copy of the Buffer RAM Address. As long as the address has not reached zero, the stepping process continues. When zero is reached, CST steps one more time and then is done. For each step, ALN converts another byte.

8. ALN sends second 64-bit binary word to Register File in OPI.

#### CONVERT-TO-DECIMAL SEQUENCE

The Binary/Decimal Converter network shown in BDP 1.0 converts a string of binary bytes stored in the Buffer RAM to an equivalent string of decimal bytes. This network converts each byte from binary to decimal, adds decimal equivalent data to a running total, multiplies by 256 (decimal), and stores the result for later use. This conversion is needed by the numeric move instruction which moves a binary source field to a decimal result field. It is also needed by the decimal product and quotient instructions where the source operands are converted to binary, operated on by ALN to generate a binary result, and converted back to decimal before being stored in a result field.

Common Stage 7 originally stores binary bytes in the Buffer RAM, from right to left (least significant byte first), starting at address zero. After the bytes are stored, the Buffer RAM Address Counter sets to the address of the most significant nonzero byte. This eliminates any leading zeros from the conversion. The counter decrements for each byte from this address to zero when the last byte is in the Buffer RAM. One BDP micrand causes the binary data to be read from AC, processed on the A stream, stored in the Buffer RAM, and converted to decimal. The conversion data flow follows.

1. Buffer RAM sends binary byte (most significant first) through C Stream Stages 1 through 3 to address Convert to Binary/Decimal RAM.
2. Convert to Binary/Decimal RAM sends Converted Binary data (10 decimal bits) to Converter Decimal Adder.
3. Converter Decimal Adder produces running total (initially zero) by adding Converted Binary data and Multiply Product.
4. Converter Decimal Adder stores sum in Converter Result RAM.
5. Converter Result RAM sends sum to Converter Multiply by 256 RAM.
6. Converter Multiply by 256 RAM sends Multiply Product (running decimal total times decimal 256) to Converter Decimal Adder where it will be added to the next byte.
7. Steps 1 through 6 repeat until last group of 10 decimal bits is added to running decimal total. Last group is added but running total is not multiplied.

#### CONVERT-TO-DECIMAL NETWORK

Several level 3 diagrams show the hardware needed to convert a string of binary bytes stored in the Buffer RAM (3.13) to an equivalent string of decimal bytes which are sent to the A Stream Data Mux (3.5). Figure 4-44 shows the Convert to Decimal Network and contains references to the corresponding level 3 diagrams.

The following example is intended as an aid for understanding this network. Three type 10 binary bytes are converted to four type 0 decimal bytes. The type 10 binary bytes are:

0001 0000 (16 decimal)  
0010 0001 (33 decimal)  
0011 0010 (50 decimal)

The algorithm for this example is:

$$\{(0 \times 256 + 16) \times 256 + 33\} \times 256 + 50$$

This example produces a decimal result of 01057074.

The four type 0 decimal bytes are:

0000 0001 (01 decimal)  
0000 0101 (05 decimal)  
0111 0000 (70 decimal)  
0111 0100 (74 decimal)

The Convert-to-Decimal Network goes through several iterations to perform the paper-and-pencil arithmetic shown in figure 4-45. The following steps and figure 4-44 describe the data flow for this example.



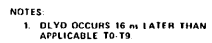


Figure 4-44. BDP Convert-to-Decimal Network

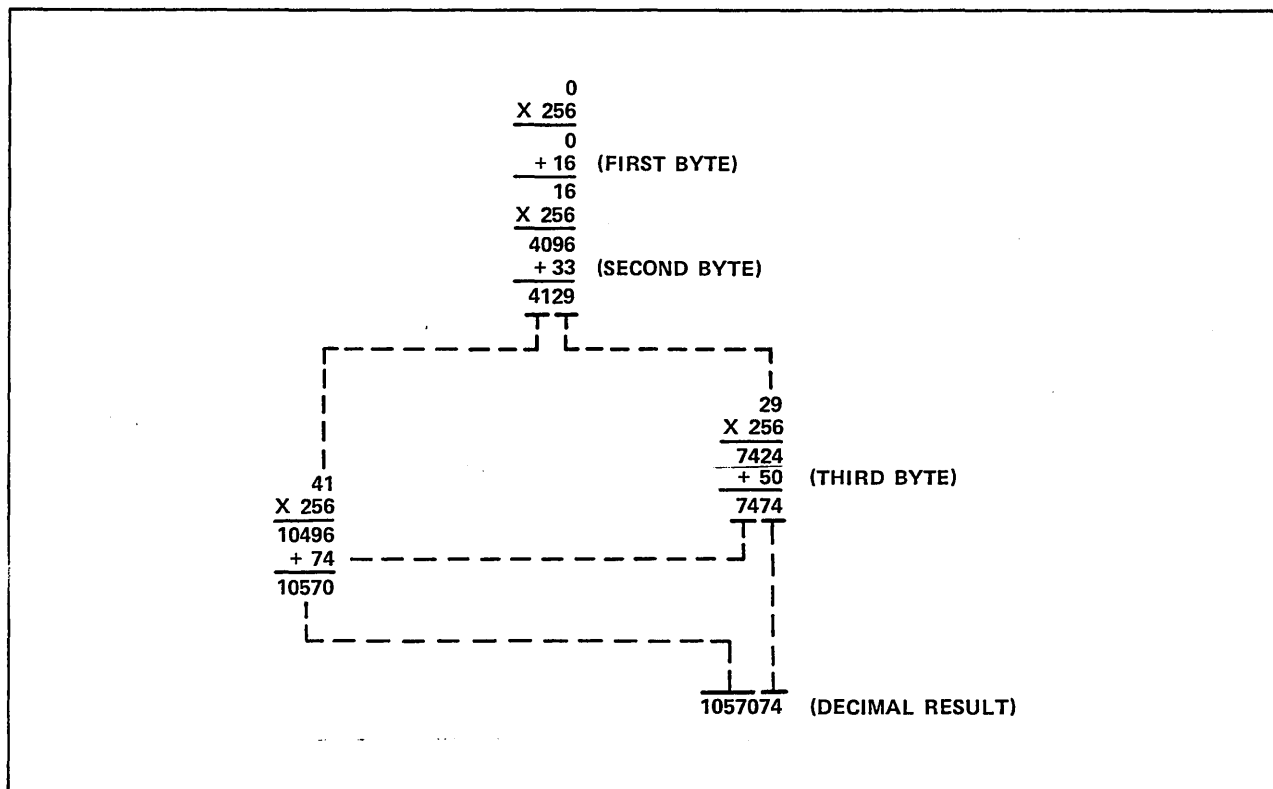


Figure 4-45. Convert-to-Decimal Example Arithmetic

#### FIRST BYTE FROM BUFFER

1. Buffer RAM sends first (most significant) binary byte 0001 0000 (16 decimal) through C Stream stages 1 through 3 to address Convert to Binary/Decimal RAM.
2. Convert to Binary/Decimal RAM sends Converted Decimal 016 (00 0001 0110 binary) to 4 1/2 Digit Decimal Adder (Adder).

Converter Multiply By 256 RAM (X256 RAM) sends Multiply Product 00000 (decimal) to Adder. (X256 RAM is initially at address 00.)

3. Adder forms 00016 (decimal) and loads it into Holding Register.
4. Holding Register sends Most Significant 2 1/2 Digits 000 back to Adder input.  
Holding Register sends Least Significant 2 Digits 16 to Converter Result RAM - Even Bytes (Even RAM).

X256 RAM continues to send Multiply Product 00000 to Adder.

5. Adder forms 00000 and loads it into Holding Register.
6. Converter Result RAM Data Output Mux (Output Mux) reads 16 from Even RAM and sends it to address X256 RAM.
7. Holding Register sends Least Significant 2 Digits 00 to Converter Result RAM - Odd Bytes (Odd RAM).

#### SECOND BYTE FROM BUFFER

1. Buffer RAM sends second binary byte 0010 0001 (33 decimal) through C Stream Stages 1 through 3 to address Convert to Binary/Decimal RAM.
2. Convert to Binary/Decimal RAM sends Converted Decimal 033 (00 0011 0011 binary) to Adder.  
  
X256 RAM sends Multiply Product 04096 (decimal) to Adder. (X256 RAM is at address 16.)
3. Adder forms 04129 and loads it into Holding Register.
4. Output Mux reads 00 from Odd RAM and sends it to address X256 RAM.
5. Holding Register sends Most Significant 2 1/2 Digits 041 back to Adder input.  
  
Holding Register sends Least Significant 2 Digits 29 to Even RAM.  
  
X256 RAM sends Multiply Product 00000 to Adder. (X256 RAM is at address 00).
6. Adder forms 00041 and loads it into Holding Register.
7. Holding Register sends Most Significant 2 1/2 Digits 000 back to Adder input.  
  
Holding Register sends Least Significant 2 Digits 41 to Odd RAM.  
  
X256 RAM continues to send Multiply Product 00000 to Adder.
8. Adder forms 00000 and loads it into Holding Register.
9. Output Mux reads 29 from Even RAM and sends it to address X256 RAM.
10. Holding Register sends Least Significant 2 Digits 00 to Even RAM.

#### THIRD BYTE FROM BUFFER

1. Buffer RAM sends third binary byte 0011 0010 (50 decimal) through C Stream stages 1 through 3 to address Convert to Binary/Decimal RAM.
2. Convert to Binary/Decimal RAM sends Converted Decimal 050 (00 0101 000 binary) to Adder.  
  
X256 RAM sends Multiply Product 07424 to Adder. (X256 RAM is at address 29.)
3. Adder forms 07474 and loads it into Holding Register.
4. Output Mux reads 41 from Odd RAM and sends it to address X256 RAM.
5. Holding Register sends Most Significant 2 1/2 Digits 074 back to Adder input.  
  
Holding Register sends Least Significant 2 Digits 74 to Even RAM. (This is fourth most significant result byte.)  
  
X256 RAM sends Multiply Product 10496 to Adder. (X256 RAM is at address 41.)
6. Adder forms 10570 and loads it into Holding Register.

7. Output Mux reads 00 from Even RAM and sends it to address X256 RAM.

8. Holding Register sends Most Significant 2 1/2 Digits 105 back to Adder input.

Holding Register sends Least Significant 2 Digits 70 to Odd RAM. (This is third most significant result byte.)

X256 RAM sends Multiply Product 00000 to Adder. (X256 RAM is at address 00.)

9. Adder forms 00105 and loads it into Holding Register.

10. Holding Register sends Most Significant 2 1/2 Digits 001 back to Adder input.

Holding Register sends Least Significant 2 Digits 05 to Even RAM. (This is second most significant result byte.)

X256 RAM continues to send Multiply Product 00000 to Adder.

11. Adder forms 00001 and loads it into Holding Register.

12. Holding Register sends Most Significant 2 1/2 Digits 000 back to Adder input.

Holding Register sends Least Significant 2 Digits 01 to Odd RAM. (This is first most significant result byte.)

X256 RAM continues to send Multiply Product 00000 to Adder.

13. Adder forms 00000 and loads it into Holding Register.

14. Holding Register sends Least Significant 2 Digits 00 to Even RAM.

#### CONVERTED DECIMAL TO A STREAM

The Output Mux sends each Converted Decimal byte twice to the A Stream Data Mux: first with the digits transposed and then normally. This is because the A stream expects unpacked data, but the Even and Odd RAMs contain packed data. The A stream cannot pause to unpack a packed decimal byte. Therefore, each byte must be sent twice so that each digit can be placed in the lower digit position. The A Stream Data Mux places a zero in the upper digit position. Data transfer is from right to left (least significant byte first) as follows.

<u>RAM</u>	<u>Output Mux</u>	<u>A Stream</u>
Even	47	07
Even	74	04
Odd	07	07
Odd	70	00
Even	50	00
Even	05	05
Odd	10	00
Odd	01	01

## SECTION 5

### VIRTUAL MEMORY



---

This section describes virtual memory which includes Segment Map (SM), Local Memory (LM), Central Memory Control (CMC), Central Memory (CM), and external mass storage. SM provides conversion from process virtual address (PVA) to system virtual address (SVA) and performs access validity testing. LM provides conversion from SVA to real memory address (RMA) and contains a high-speed cache memory. CMC controls access to CM and a Common Memory from four ports. CM contains up to 16 million bytes (2 million words). External mass storage contains the remaining data in virtual memory.

Each executing procedure operates in its own address space which is divided into segments. Each of up to 4096 segments may be 2.1 billion bytes (262 million words) long. The segment is the unit of virtual memory management. A segment contains a variable number of pages. The page is the unit of real memory management. Each page contains a variable number of words. Page size is established when the system is initialized. The standard for C180 is 512 words per page, and up to 41 million pages per segment. Data transfers between CM and external mass storage a page at a time.

#### VIRTUAL MEMORY ADDRESS CONVERSION

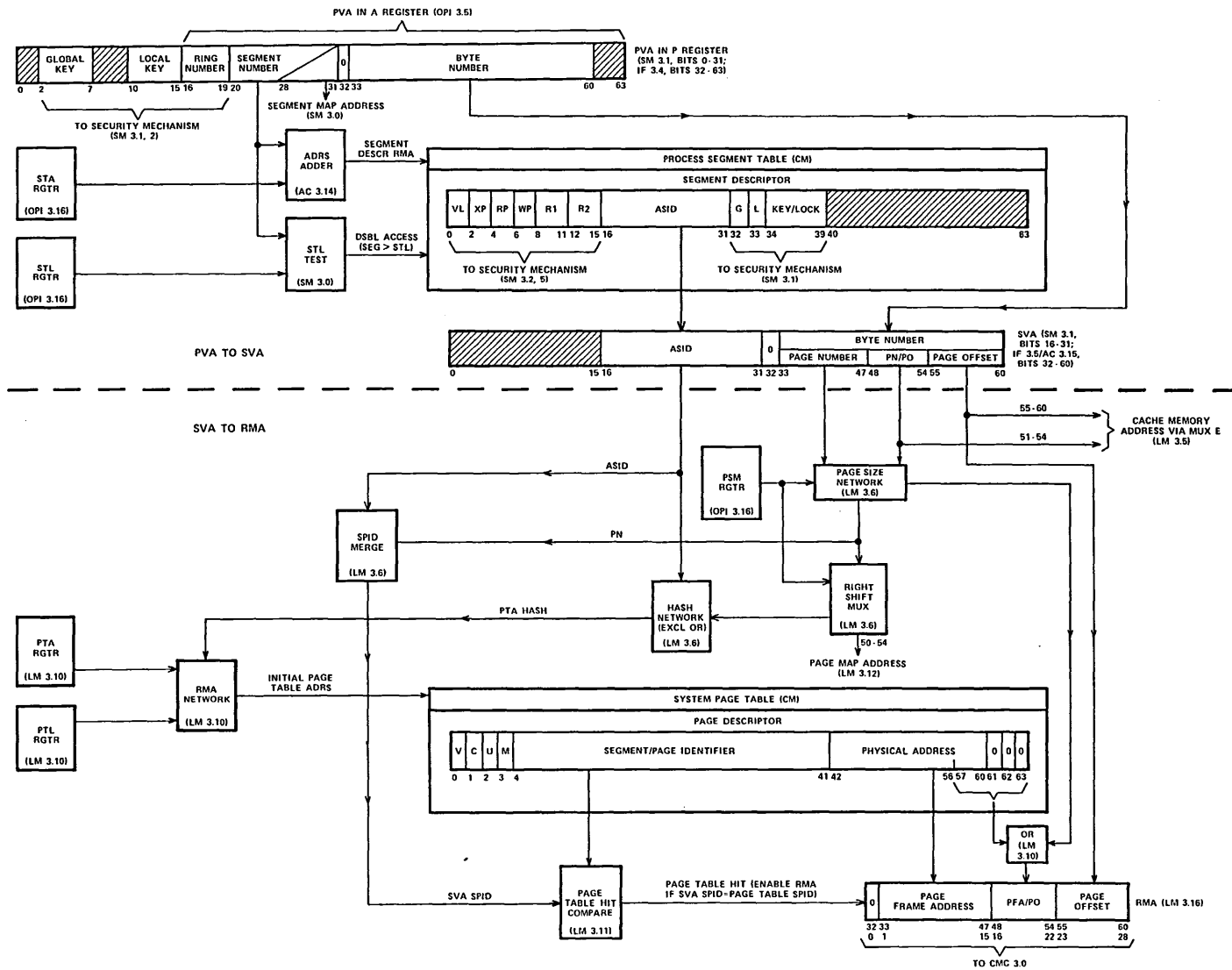
The address conversion network shown in figure 5-1 translates a PVA into an RMA in two steps. The first step translates a PVA to an SVA, using the Process Segment Table. The SVA is the global address used to identify each page in the System Page Table. The second step translates an SVA to an RMA, using the System Page Table. The RMA is invisible to the programmer.

#### BASIC CONVERSION COMPONENTS

The following paragraphs describe the basic conversion components shown in figure 5-1.

#### Segment Table Address and Length Registers

The Segment Table Address (STA) and Segment Table Length (STL) values are part of each process's exchange package. These values are loaded into registers in OPI. The Segment Number times eight (to convert to bytes) plus STA (byte address) forms the Segment Descriptor's RMA byte address in CM. STL is the user's table length. It is compared with the Segment Number for invalid segment testing.



### Figure 5-1. Virtual Memory Address Conversion



### Process Segment Table

System software creates this table in CM for each process. Table bits 16 through 31 contain an Active Segment Identifier (ASID) which uniquely identifies each active segment on a system-wide basis. Other parameters are described later under Virtual Memory Address Security.

### Page Table Address, Length, and Page Size Mask Registers

System software loads the Page Table Address (PTA), Page Table Length (PTL), and Page Size Mask (PSM) Registers during system initialization. PTA and PTL also load into live registers in LM. PSM also loads into a live register in OPI and is sent to LM. These registers, along with the Hash Network, form the initial entry address into the System Page Table for each page table search sequence.

### System Page Table

System software creates this table in CM which contains a Page Descriptor for each active page located in CM. Each descriptor contains control fields, Segment/Page Identifier (SPID), and a Physical Address.

Page descriptor control fields V, C, U, M specify:

<u>Field</u>	<u>Bit</u>	<u>Set</u>	<u>Clear</u>
V	0	Valid Entry	Invalid Entry
C	1	Continue Search	Stop Search
U	2	Used	Unused
M	3	Modified	Unmodified

The SPID (bits 4 through 41) identify the SVA for the Page Descriptor entry. It includes the 16-bit ASID and the 22-bit page number. For systems in which the page size is larger than 512 bytes (page number less than 22 bits), zeros must fill the corresponding lower (rightmost) bit positions.

The Physical Address (bits 42 through 63) becomes the Page Frame Address (leftmost RMA bits). If page size is larger than 512 bytes, zeros fill the corresponding lower (rightmost) bit positions.

### PVA TO SVA CONVERSION

SM converts PVAs from the P Register (for addressing instructions) and A Register (for load/store instructions) to SVAs which are sent to LM. Only the Segment Number and the Byte Number are used for PVA to RMA conversion. The Global and Local Keys and Ring Number are used by the Security Mechanism. To allow CM read accesses on a word boundary, the lower (rightmost) three bits of the Byte Number are disregarded. For byte instructions, Address Control (AC) retains these three bits to address the desired byte when the word is read from memory or forms mark bits when the word is written to CM.

As shown in figure 5-1, the Segment Number times eight plus STA addresses the Process Segment Table in CM. If the Segment Number is greater than STL, the CM reference is not made and the process is interrupted. The Process Segment Table contains Segment Descriptors. Every user has a separate table which is built for the user by System Software. The addressed Segment Descriptor contains an Active Segment Identifier (ASID) and some security parameters. The ASID joins the Byte Number from the PVA to form the SVA. The SVA goes to LM to address Cache Memory (organized on an SVA basis) and for conversion to RMA.

#### SVA TO RMA CONVERSION

LM converts SVAs to RMAs which are sent to CMC to address a word in a page. The System Page Table in CM contains the starting address of each page currently residing in CM. Although page size is a variable, pages currently in use and defined by software have a fixed length of 512 words. Figure 5-1 and the following paragraphs describe the SVA to RMA conversion.

1. Page Size Network determines Page Number and Page Offset from Byte Number in SVA based on contents of Page Size Mask Register. This register contains seven bits. If all are clear, Byte Number bits 48 through 60 all become Page Offset bits. If all are set, bits 55 through 60 become Page Offset bits. Currently, the operating system uses a 512-word page size. Therefore, a mask of 1111000 selects bits 52 through 60 as fixed Page Offset bits.
2. Right Shift Mux shifts Page Number so that the least significant Page Number bit is in bit position 54. The amount of right shift is determined by the page size. If page size is 512 words, the right shift is three places because bits 52 through 54 are part of the Page Offset.
3. Hash Network performs an exclusive OR of shifted page number and ASID from SVA. This generates PTA Hash which forms a random address into the System Page Table. System software uses a similar hashing algorithm to place the entry in the table.
4. RMA Network forms address for entry in variable-length System Page Table. PTA Hash is the variable portion of the Initial Page Table Address. The PTA Register provides the fixed upper bits while the PTL Register determines how many are fixed. The PTL Register is an eight-bit mask. If all mask bits are clear, all 19 bits of PTA are fixed which provides a PTL of 512 entries. If all bits are set, only the upper 11 bits are fixed which provides 131K entries. Bits 60 through 63 are zeros; therefore, each initial PTA is to an even word.
5. Page Table Hit Compare determines whether or not addressed Page Descriptor is the correct one. A Page Table Hit occurs when SPID in SVA is equal to SPID in a valid Page Descriptor.
6. If a Page Table Hit occurs, the Physical Address in the Page Descriptor becomes the Page Frame Address. This joins with the Page Offset in the SVA to become the RMA. The upper part of the Page Offset comes from the Page Size Network; the number of bits depends on the value of PSM. Currently, bits 52 through 54 come from PSM (page size equals 512 words).
7. If a Page Table Hit does not occur, the table is searched. The operating system builds the table so that Initial Page Table Address will hit or be close to the correct entry. If the page is not in CM, the search stops after 32 consecutive entries or if the continue bit is clear. The searching mechanism goes from entry to entry if the continue bit is set. The continue bit indicates that two or more entries have the same hash address in the table. This is normal. The valid bit indicates the corresponding page is in CM. The valid bit must be set and a SPID match must occur to achieve a Page Table Hit. If a page table search does not find a valid entry, system software fetches the page from external mass storage and updates the System Page Table.

## MAP CONCEPTS

To minimize the number of CM references required, the CPU uses three high-speed RAMs: the SM RAM, the Page Map RAM, and the Cache Memory RAM. The SM RAM contains up to 32 of the most recently used Segment Descriptors from the Process Segment Table. In the first stage of address translation, this RAM translates the PVA to an SVA. This SVA is then sent to the Cache Memory RAM and Page Map RAMs in LM. Cache Memory contains up to 4,096 of the most recently used words in system virtual memory, while the Page Map contains up to 128 of the most recently used Page Descriptors from the System Page Table. Simultaneously, the hardware tests the Cache Memory and Page Map to see if the SVA is present. If a Cache Memory hit occurs, then no further action is required because the CPU reads the desired data from Cache Memory. If the desired data is not in Cache Memory, then the Page Map test is relevant. If a Page Map hit occurs, the second stage of address conversion (SVA to RMA) is completed, and the CPU reads a 4-word block of data from CM. If no Page Map hit occurs, the CPU initiates a search of the System Page Table. These RAMs are described in detail later.

## VIRTUAL MEMORY ADDRESS SECURITY

The security mechanism in SM provides controlled access to all code and data. This protects the operating system from the users, the users from each other, and the users from the operating system. The basic element of protection is the user's address space, which is the set of addresses assigned to an executing procedure by the operating system. It is defined by the set of Segment Descriptors contained in the Process Segment Table shown in figure 5-1 and described earlier. Each entry in this table defines the security protection features for one segment. One way a user can attempt to access a segment not in his address space is to exceed the STL, which causes his procedure to be interrupted. Another way is for a user to attempt to use a PVA for an invalid segment (user's Process Segment Table valid field equals zero). This also causes an interrupt.

Once a user has been confined to an address space, the segment becomes the basis of the security mechanism. There are three forms of protection within a user's address space.

- Segment Access Privileges.
- Ring Address Protection.
- Key/Lock Address Protection.

For every CM access attempted, all three of these tests must be successful. If any of them fails, an access violation interrupt results and the user is interrupted by system software. SM conducts the security tests at the same time it converts PVA to SVA. The three forms of protection are described in the following paragraphs.

## SEGMENT ACCESS PRIVILEGES

The following paragraphs describe segment types and segment descriptor access control fields.

### Segment Types

There are three types of segments: execute, data, and binding section. Execute segments contain instructions and data which are fetched but not read or written. Data segments contain operands which may be read and/or written. Binding section segments contain pointers to other segments. The binding section is used during call instructions. The binding section points to the addresses of the segments being accessed. Some segments are defined as cache bypass segments. These segments never reside in Cache Memory (for example, the exchange package).

### Segment Descriptor Access Control Fields

The first four fields of the Segment Descriptor (figure 5-1) define the access privileges for the segment. The fields are described as follows.

#### Valid (VL) Field

- 00 Invalid entry.
- 01 Invalid entry (reserved).
- 10 Regular segment. This is an active segment for the executing procedure.
- 11 Cache bypass segment. This segment does not reside in Cache Memory.

#### Execute Privilege (XP) Field

Data in an execute segment can be read-accessed by a load relative instruction.

- 00 Nonexecutable segment.
- 01 Nonprivileged executable segment. This segment has not been granted either local or global execution privileges.
- 10 Local-privileged executable segment. This segment has been granted local execution privilege.
- 11 Global-privileged executable segment. This segment has been granted global and local execution privileges.

### Read Privilege (RP) Field

RP values of 01, 10, and 11 also require that validity and ring checks are successful to allow access.

- 00 Nonreadable segment.
- 01 Read under control of key/lock. This segment is readable only if PVA Keys in P Register match Segment Descriptor Lock.
- 10 Read not under control of key/lock. This segment is readable by any instruction.
- 11 Binding section. This segment contains pointers to other segments, is not under control of key/lock, and is used by call instructions.

### Write Privilege (WP) Field

WP values of 01 and 10 also require that validity and ring checks are successful to allow access.

- 00 Nonwritable segment.
- 01 Write under control of key/lock. This segment is writable only if PVA Keys in P Register match Segment Descriptor Lock.
- 10 Write not under control of key/lock. This segment is writable by any instruction.
- 11 Nonwritable segment (reserved).

### RING ADDRESS PROTECTION

Each address space is organized into a maximum of 15 rings of protection. This provides 15 different levels of security for program execution. The ring protection mechanism is controlled by Segment Descriptor fields R1 and R2, code base pointer field R3, and PVA Ring Numbers. Execute and data segments may be accessed from more than one ring. R1 and R2 define the limits of these accesses.

### Ring Hierarchy

The rings are organized hierarchically. Ring 1 has the highest privilege, ring 15 the lowest. A process designated by a given ring number can read and write segments with its own ring designation and any higher numbered ring. This process can call procedures designated by its own ring number and any lower numbered ring, provided that the called procedure's R3 value permits the access.

### Accessor Ring Number

An execute segment is accessed by a PVA from the P Register (for relative call and relative branch instructions) or the A Register (for call, return, and intersegment branch instructions). A data segment is accessed by a PVA from the A Register. Trap and debug pointers also contain accessor ring numbers. The Ring Number is in bit positions 16 through 19. Figure 5-1 shows the PVA format.

### Segment Descriptor Ring Fields

The accessee is the Segment Descriptor shown in figure 5-1. This entry from the Process Segment Table contains an R1 field in bit positions 8 through 11 and an R2 field in bit positions 12 through 15. R1 and R2 define ring access privileges for execute, read, and write operations in the segment.

### Code Base Pointer Ring Limit

Code base pointers provide the linkage between executable segments during call instructions and trap sequences. These pointers reside in a special segment called a binding section. They provide the callee's starting instruction address to the caller. The caller's Ring Number in the A Register, used to access the code base pointer in the binding section, must be equal to or less than the code base pointer R3 limit in bit positions 12 through 15. The code base pointer ring number in bit positions 16 through 19 can be any value. However, a zero value indicates the code base pointer is unlinked and causes the process to be interrupted. The code base pointer bits are described as follows.

<u>Bits</u>	<u>Description</u>
0-3	Not Used
4-7	Virtual Machine Identifier
8	External Procedure Flag
9-11	Not Used
12-15	R3 Limit
16-19	Ring Number
20-31	Segment Number
28-31	Map Address
32-63	Byte Number (First Instruction)

### Ring Definitions

The following paragraphs describe ring functions during execute, read/write, and call accesses.

#### Execute Access

A segment may be executed by several procedures designated by different ring numbers. The segment's execute ring bracket is defined by Segment Descriptor fields R1 and R2. If the P Register Ring Number is equal to or greater than R1 and equal to or less than R2, the procedure can execute this segment.

An execute segment is accessed by call, return, and intersegment branch instructions and exchange and trap interrupts. Upon entry into an execute segment, the upper 32 bits of the new program address (P), as determined by the operation, are latched in SM. P (upper, leftmost) remains unchanged during execution in the segment. Thus, the Ring Number remains unchanged until another instruction issues or an interrupt occurs. During an inward call instruction, the P Register Ring Number changes to the value of the called segment ring bracket R2.

### Read/Write Accesses

An executing procedure can read from a data segment if the segment's PVA Ring Number is equal to or less than the Segment Descriptor R2. It can write a segment if the segment's PVA Ring Number is less than R1. If R1 and R2 are 10 and 12, this segment can be read using a PVA with Ring Number of 12 or less and written using a PVA with Ring Number of 10 or less.

### Call Access

A procedure in one execute segment can call a procedure in another execute segment using a code base pointer. This pointer comes from a binding section segment which is addressed by a PVA in the A Register.

SM makes two tests before the call is made. First, the call must be within the same ring or from an outward ring to an inward ring. This test passes if the PVA Ring Number from the A Register is equal to or greater than Segment Descriptor R1. The second test ensures that the caller has adequate privileges to make the call. This test passes if the PVA Ring Number from the A Register is equal to or less than the code base pointer R3 limit. Therefore, the called procedure can block calls from procedures with a ring designation greater than the R3 limit.

### Multiring Segment Example

An example of the four ring limits/brackets is shown in figure 5-2. The R1, R2, and R3 values define the ring limits/brackets for the example segment as follows:

R1 = 3  
R2 = 5  
R3 = 7

R1 and R2 are contained in the Segment Descriptor for this example. R3 is an attribute of a code base pointer. While R1 and R2 together constitute an execute bracket; R1, R2, and R3 singly provide write, read, and call limits, respectively. Writes require A Register Ring Number equal to or less than R1; reads require A Register Ring Number equal to or less than R2.

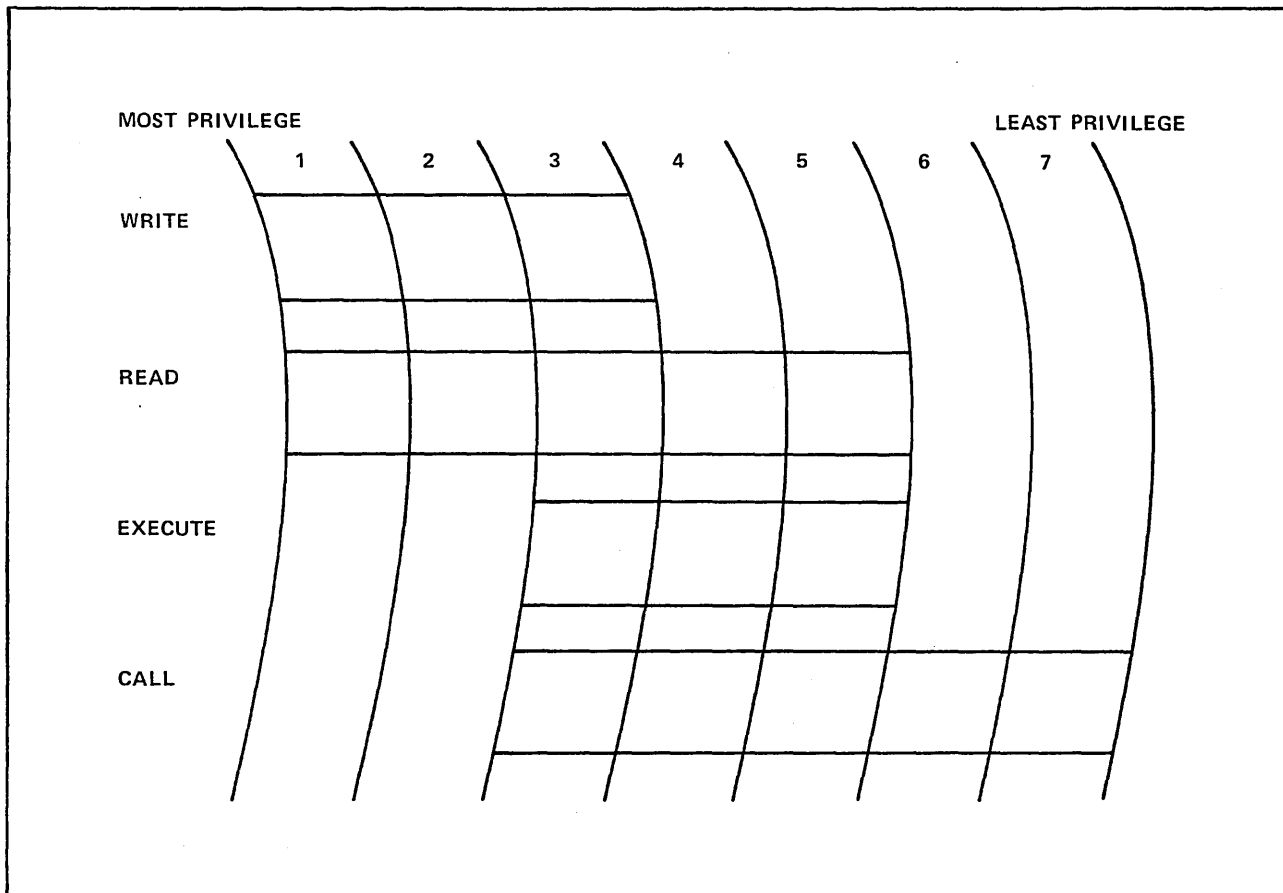


Figure 5-2. Multiring Segment Example

#### Write

The segment may be written by a procedure from another segment whose ring number is 1 through 3.

#### Read

The segment may be read by a procedure from another segment whose ring number is 1 through 5.

#### Execute

The segment may be executed by a procedure whose ring number is 3 through 5. If the segment is called by a procedure whose ring number is 3, the procedure will execute with ring number 3. If the segment is called by a procedure whose ring number is 4, the procedure will execute with ring number 4, and so on. If called by a procedure whose ring number is 6 or 7, the procedure will execute with ring number 5 - the least privileged of ring number 3 through 5.



## Call

The segment may be called by a procedure whose ring number is 3 through 5 (execute) or 6 and 7 (call). The segment cannot be called by a procedure whose ring number is 1 or 2 because this would be an outward call which is illegal. It cannot be called by a procedure whose ring number is 8 through 15 because this exceeds the call limit.

### Ring Usage Example

An example of ring usage is shown in figure 5-3. The rings are denoted in parentheses as: (R1, R2, R3). The dashed lines indicate logical extensions of the data segment. The execute segments are not extended because the R1 and R2 fields restrict execution to a single ring. The heavy arrows indicate inward calls.

On the left side of figure 5-3 is an execute segment with R1 equal to 3 and R2 equal to 11. This procedure may be executed with ring numbers of 3 through 11. This procedure could be the FORTRAN math library. A procedure with ring number 11 may call on Square Root which would execute with ring number 11. Similarly, a procedure with ring number 8 would execute Square Root with ring number 8. Therefore, the Square Root procedure always executes with the privilege of the caller.

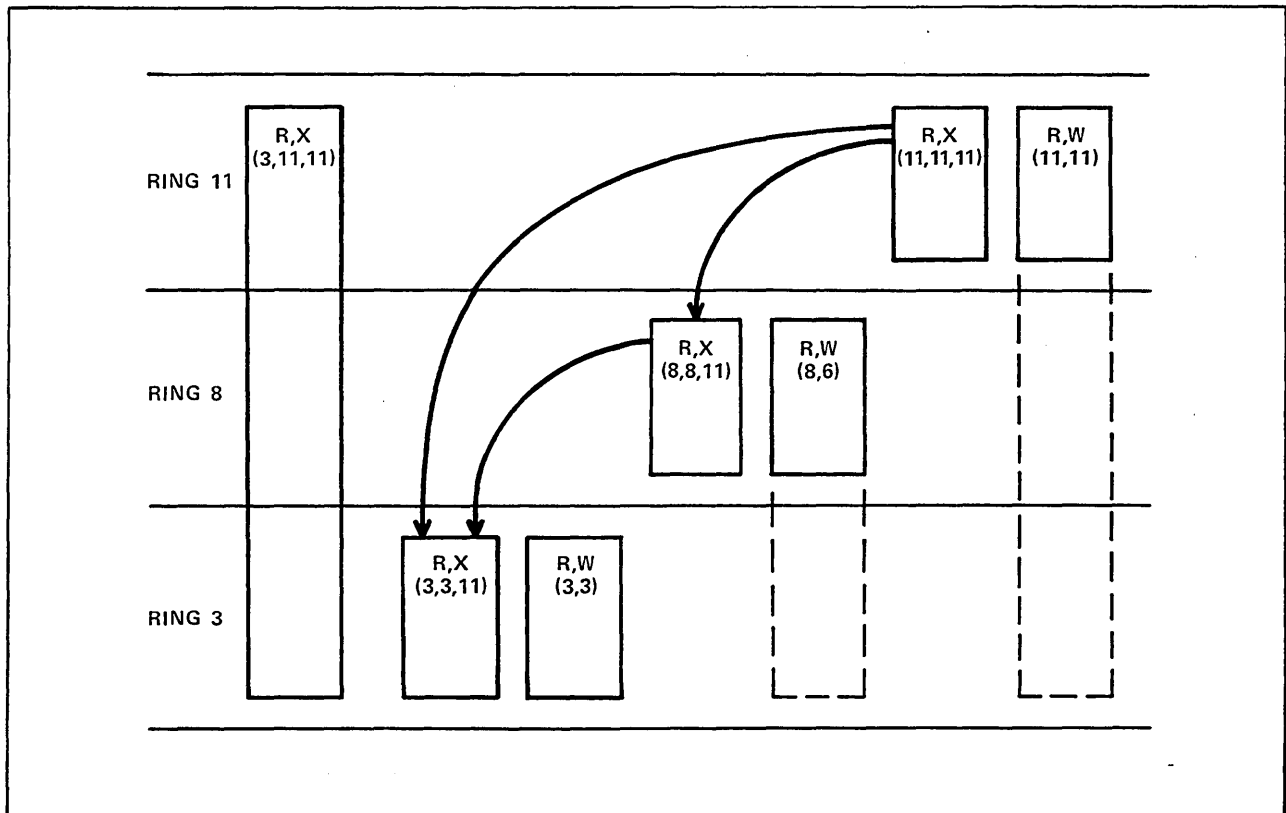


Figure 5-3. Ring Usage Example

The procedure with ring number 11 may call on the procedure with ring number 8 whose call limit (R3) is 11.

The procedure with ring number 8 may read and write the data segment belonging to the procedure with ring number 11. This is because the segment has read and write access privileges and R1 and R2 are both 11. Thus, the procedure with ring number 11 may pass operands to and receive results from the procedure with ring number 8.

The procedure with ring number 3 may be called from either procedure with ring number 8 or 11. This is because the procedure with ring number 3 has its call limit (R3) set to 11.

#### KEY/LOCK ADDRESS PROTECTION

SM conducts key and lock tests simultaneously with the segment access and ring tests. Local keys and locks protect private data segments from other executing procedures. Global keys and locks can isolate applications from each other in the same ring of protection. Unlike rings, keys and locks have no hierarchical significance. All that matters is whether or not the keys and locks are equal. Access is granted if the keys and locks are equal, the key is a master key, or the lock is unlocked.

#### P Register Key Field

The first two fields in the P Register (figure 5-1) contain Local and Global Keys. For a data segment access or return operation, both Local and Global Keys are compared with the Segment Descriptor Lock. For a call or intersegment branch, only the Global Key is compared with the lock. A local or global field of zero denotes a master key.

#### Segment Descriptor Key/Lock Fields

The last three Fields in the Segment Descriptor (figure 5-1) contain G, L, and Key/lock. There is a 6-bit lock associated with every segment. Whenever an execute segment is entered, that is, by an exchange, its lock becomes the current key and is placed in the P Register. The G and L fields describe the contents of the Key/Lock field for procedure (execute) and data (read/write) segments as follows.

<u>G/L</u>	<u>Execute Segment</u>	<u>Read/Write Segment</u>
G = 0	Global master key or unlocked	Global unlocked
G = 1	Global key or lock	Global lock
L = 0	Local master key	Local unlocked
L = 1	Local key	Local lock

### Key/Lock Applications

A master key fits any lock and any key fits an unlocked segment. Access to one segment from another segment is granted if the first segment has a master key, the second segment has no lock, or if the keys match the locks.

For read/write access, SM performs tests selectively as controlled by the RP and WP fields in the Segment Descriptor. Even though a global and/or local lock is specified for a segment, the test for read access is made only if RP equals 01. Similarly, the test for write access is made only if WP equals 01.

For calls and returns, SM performs global key/lock tests. A call is permitted when caller has a master key, called procedure (callee) has no lock, or key matches lock. The callee assumes the caller's global key unless the callee had a master key and the caller did not. However, the caller always assumes the callee's local key. On a return, SM ensures that caller's new global key (from stack frame save area in CM) matches caller's global lock (from Segment Descriptor), or that global lock is unlocked. The caller's new local key must match caller's local lock.

### Key/Lock Usage Example

An example of key/lock usage is shown in figure 5-4. In this example, three rings of protection are used. Ring 3 is the most privileged ring, where parts of the operating system reside. Ring 8 contains two applications which must be isolated from each other, but are callable from a user in ring 11. The information in parentheses defines the global and local locks and whether they apply to read accesses (R), write accesses (W), or both. The heavy arrows indicate inward calls.

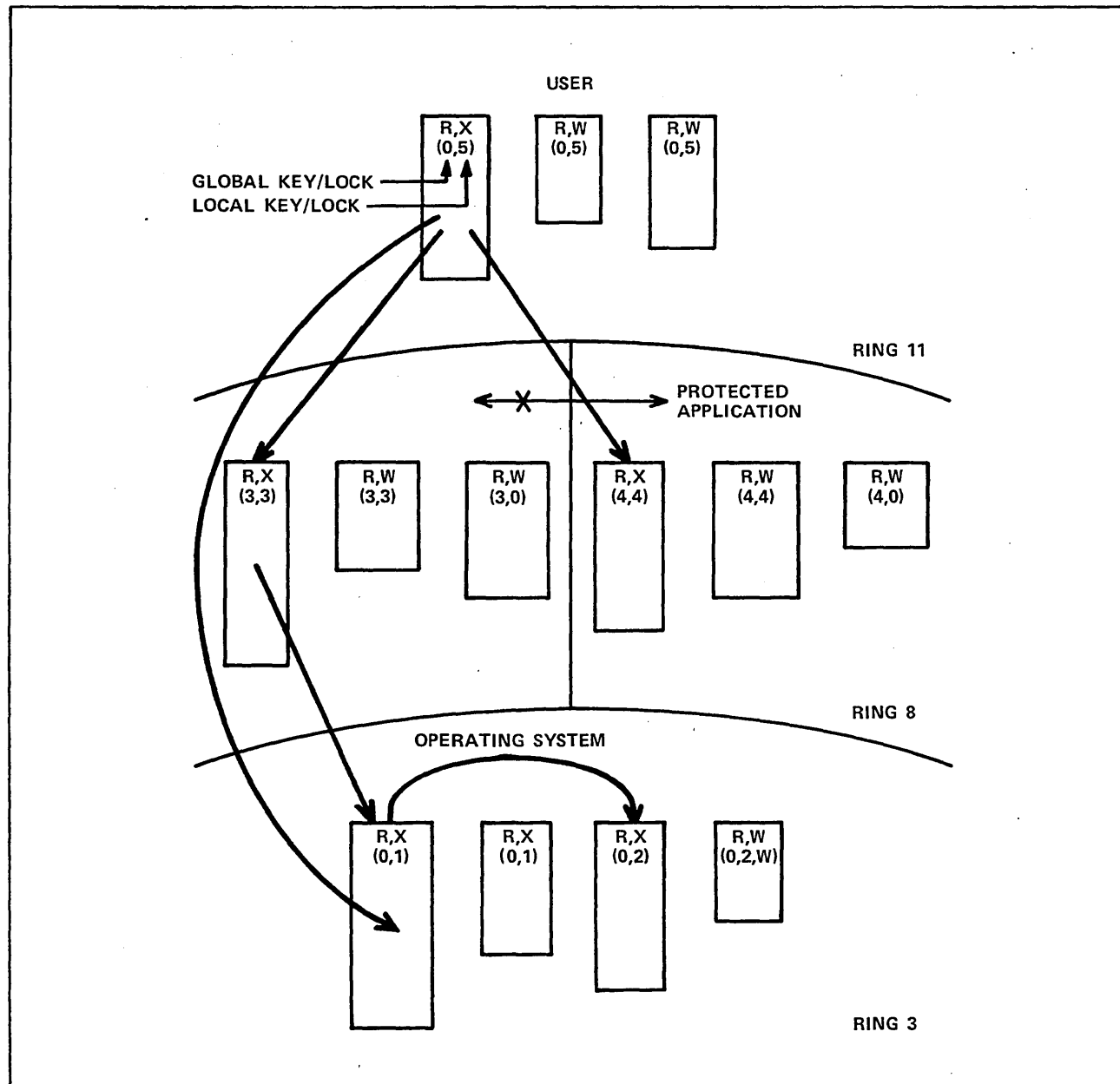


Figure 5-4. Key/Lock Usage Example

To isolate the two applications in Ring 8 from each other, they are assigned different global key/lock values. The two applications cannot call each other or read each other's data segments. They have been totally isolated from each other even though they are in the same ring. For the applications to call Ring 3 and be callable from Ring 11, the operating system and user are assigned Global Key/Lock values of 0 (master keys, no locks). Therefore, the user may call either application or the operating system, because the user has a master global key. When the user calls an application, the application runs with its own global key. This ensures continued isolation. When the call returns, the user's global key is restored to him.

When a call is made to a procedure in another segment, callee executes with its own local key. This value is associated with the local lock of the data segments accessed by that procedure. In the example, the user has two read/write data segments each with Local Key/Lock value of 5. Consequently, the applications in Ring 8 and the operating system in Ring 3 cannot access the data segment because their local locks are not equal to 5. Similarly, the operating system cannot access either of the application's data segments. Local Key/Locks protect local data regardless of ring brackets in use.

#### SEGMENT MAP

SM converts PVAs to SVAs and performs address security tests. SM performs these two functions simultaneously under micrand control.

#### MICRAND CONTROL

CST sends both the SM and LM micrand fields to SM. Both fields are part of the AC micrand which is described in section 3. The 16-bit SM micrand field controls SM operations. SM sends the 11-bit LM field to LM unchanged.

#### PVA TO SVA CONVERSION

SM converts PVAs to SVAs (shown earlier in figure 5-1) using SM RAMs to store Segment Descriptors associated with the most recently accessed segments. Refer to SM 1.0.

#### Segment Map RAMs

SM uses a set associative technique to convert a Segment Number from an A or P Register to an ASID. Set associative means that two RAM sets are addressed simultaneously by the Segment Number. Each set contains a tag and associated data. A tag formed from the Segment Number must then match a tag from one of the addressed RAMs. Figure 5-5 shows the SM RAM formats. Each RAM contains up to 16 most recently used Segment Descriptors from the process segment table in CM. The lower four bits of the Segment Number address the SM RAM Sets 0, 1, and the Map Status RAMS. The Map Hit/Miss Control compares the upper eight bits of the Segment Number with the SM RAM Tag field (upper eight bits of the associated Segment Number) from the addressed location. If they match and the SM RAM entry is valid, the ASID in this Segment Descriptor is sent to LM or AC for the SVA to RMA conversion. When an execute segment is initially entered, the ASID contained in the Segment Descriptor is latched into the P Descriptor Save Register. This ASID joins a byte address from IF to become the IF SVA. Subsequent instruction requests within the segment continue to use the output of the P Descriptor Save Register as the IF ASID. The ASID, which becomes part of the AC address to access operands or write results in memory, enters AC directly from SM RAM set 0 or 1. If the required Segment Descriptor is not in either RAM, the Segment Map Miss signal initiates a sequence to fetch the Segment Descriptor from the process segment table in CM. The Segment Descriptor Fetch sequence is described later.

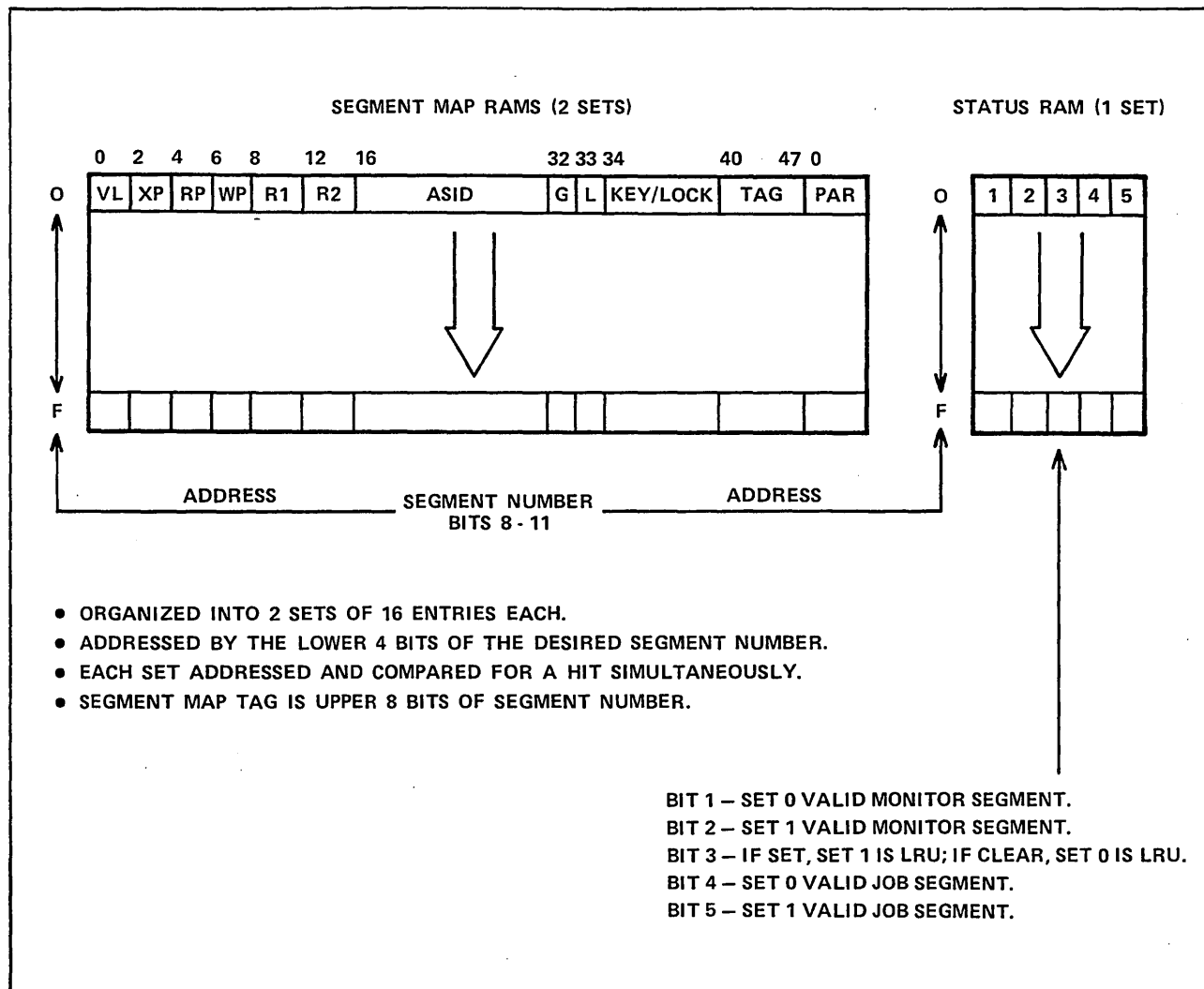


Figure 5-5. Segment Map RAM Format

### PVA Register

This register is typically loaded with a PVA from OPI. For data segment access, the PVA originates in an A Register. AC receives the byte number; SM receives the Ring and Segment Number. For execute segment access, the PVA may originate from the P Register location of an exchange package. The A Register contains the address of the called or branched to segment. For execute segment access, the PVA may also originate from the code base pointer from a binding section segment. The code base pointer links an execute segment during a call instruction.

### Segment Descriptor Input Register

This register is loaded with either STA or the Segment Descriptor from OPI. During an exchange operation into job mode, the new STA from OPI is compared with STA from the previous job process. If they are different, indicating a different job's process segment table is being used, Purge Control invalidates all job mode entries in the RAM.

During a segment descriptor fetch sequence, this register is loaded with the Segment Descriptor from the process segment table. The Segment Descriptor combines with the PVA Segment Number and enters the SM RAM. The new descriptor enters either set 0 or 1 at the location specified by the lower four bits of the Segment Number (Map Address). The receiving set is selected by the allocation algorithm described below.

### Segment Map RAM Allocation

Map Hit Control and Map Status RAMs determine which set receives the fetched Segment Descriptor entry.

- If one set is invalid, it receives the new entry.
- If both sets are invalid, set 0 receives the new entry.
- If in job mode and one set contains a valid job entry and the other contains a valid monitor entry, the set with a monitor entry receives the new entry.
- If in monitor mode and one set contains a valid monitor entry and the other contains a valid job entry, the set with a job entry receives the new entry.
- If in job mode and both sets contain valid job entries or if in monitor mode and both sets contain valid monitor entries, the least recently used (LRU) set receives the new entry. The LRU bit in the Map Status RAMs indicates which set is the LRU. If the LRU bit is set, the new entry goes into set 1. If clear, the new entry goes into set 0.

### Segment Descriptor Selection

The Segment Descriptor Mux selects one of four inputs. The Shortstop Descriptor is selected after a segment descriptor fetch sequence. The SM RAM descriptor from set 0 or 1 is selected depending on which set had a tag compare (SM hit). The remaining input comes from the SM Bypass Address Mux which bypasses the PVA to SVA conversion and uses the already converted contents of the P Descriptor Save Register.

For operand read or write operations, the Segment Descriptor Mux sends the ASID to AC, which forms the byte number part of the address.

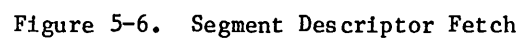
For instruction access, the Segment Descriptor Mux sends the entire descriptor to the P Descriptor Save Register. This register holds the descriptor while a procedure is executing in this segment. The contents change only after a call, return, exchange, or intersegment branch instruction. The IF ASID joins with the byte number from IF to form the IF SVA.

#### SEGMENT DESCRIPTOR FETCH SEQUENCE

This sequence occurs when Map Hit/Miss Control detects an SM Miss. Several functional areas participate in the sequence which is shown in figure 5-6 and described below.

1. Hit/Miss Control in SM sends SM Miss to LM to block LM request from ICP rank 32 and to ICP to block pipe movement, block go signal to functional units, and set Segment Descriptor Fetch FF. Micrand which caused SM Miss is held in ICP rank 32 during segment descriptor fetch sequence.
2. Segment Descriptor Fetch FF enables Segment Number in SM and Segment Table Address from OPI live registers to Address Adder in AC.
3. Address Adder Forms RMA for segment descriptor in process segment table in CM. AC sends RMA to LM.
4. Segment Descriptor Request FF in ICP sets to enable LM request for descriptor.
5. LM bypasses cache, does not convert address, and sends segment descriptor address to CMC/CM.
6. Segment Descriptor Wait FF in ICP sets to enable LM Read Data into DAI Mux Register in OPI.
7. CMC/CM sends CM read data to LM.
8. LM generates LM Response which signifies arrival of Segment Descriptor from LM to OPI. This clears Segment Descriptor Fetch FF in ICP to enable loading data from DAI Mux Register to SM RAM Set 0, 1.
9. LM Read Data passes through DAI Mux Register and enters either SM RAM Set 0 or 1 (depending on which set is allocated) at address specified by lower four bits of Segment Number. This Segment Descriptor also goes to SM Segment Descriptor Mux through a shortstop path.
10. SM Miss drops which enables ICP pipe and enables go signal to functional units. Micrand which caused SM Miss (currently in ICP rank 32) reissues go signals.
11. As a result of stopping pipe (step 1), operands needed to form Byte Number are held in OPI registers. OPI reissues these operands to AC where A/C Stream Address Counter reforms Byte Number. This Byte Number combines with ASID from SM Segment Descriptor Mux to form AC Address to LM.
12. Segment Descriptor Request and wait FFs in ICP clear.





## P REGISTER (UPPER)

The New and Old P Registers (SM 1.0) contain the upper 32 bits of the program address. The New P Register buffers information entering the Old P Register. This information includes global key, local key, and ring number which are tested by SM for access violations. A live register read operation of the P Register appends these 32 bits to the lower 32 bits from IF and sends them to OPI. The Keys and ring numbers are described in the following paragraphs.

## ADDRESS SECURITY TESTS

SM conducts four address security tests for any procedure that attempts to access a segment. These are Segment Table Length, Segment Validity, Key/Lock, and Ring. Refer to SM 1.0.

### Segment Table Length Test

The Segment Table Length Register (from the exchange package) specifies the process segment table length. If a user requests a segment number which exceeds this length (Segment Greater Than STL), the Invalid Segment Test sends SM Invalid Segment to ICC (via ICP) which initiates an interrupt.

### Segment Validity Test

The Control fields from the Segment Descriptor Mux define the access privileges for the segment. These Access Control Fields VL, XP, RP, and WP are described earlier under Segment Access Privileges. Segment Validity Test compares these fields with the type of access attempted (execute, read, write) and sends Validity Test Results to Access Violation Test. If any test fails, Access Violation Test sends SM Access Violation to ICC (via ICP) which initiates an interrupt.

### Key/Lock Test

Local keys and locks protect private data segments. Global keys and locks isolate applications in the same ring of protection from each other. Key/lock concepts are described earlier under Key/Lock Address Protection.

The Key/Lock Access Control Field from the Segment Descriptor Mux is compared with either the Old P Keys or PVA Keys depending on the operation. The Key/Lock Test compares the Key/Lock field with the Old P Keys when an executing procedure is reading and writing data segments. In this case, PVAs originate in A Registers. The Key/Lock Test compares the Key/Lock field with the PVA Keys during call, return, exchange, or intersegment branch operations. In these cases, PVAs originate in the P Register. If any test Fails, Access Violation Test sends SM Access Violation to ICC (via ICP) which initiates an interrupt.

The Selector determines which New P Keys enter the New P Register when an execute segment is entered. For example, a caller picks up callee's local key and may bring its own global key, depending on certain conditions explained earlier under Key/Lock Address Protection. Once the new keys (and ring number) are determined, the contents of New P enter Old P and remain there during execution within the segment.

### Ring Test

The 15 hierarchical rings of protection provide 15 different levels of security for program execution. Procedures designated by a given ring number can read and write segments with an equal or higher numbered ring. A procedure can call other procedures designated by an equal or lower numbered ring. Ring concepts are described earlier under Ring Bracket Address Protection.

The R1 and R2 Access Control Fields from the Segment Descriptor Mux are compared with either PVA Ring, code base pointer R3, New P Ring, or old P Ring. The Ring Test compares the R1 and R2 fields with the PVA Ring during call, exchange, or intersegment branch instructions. The Ring Test compares Old P Ring with R3 during call instructions. The Ring Test compares PVA Ring with New P Ring during return instructions. The Ring Test compares R1 or R2 fields with A Register PVA Ring when an executing procedure writes or reads a data segment. If any test fails, Access Violation Test sends SM Access Violation to ICC (via ICP) which initiates an interrupt.

The Selector determines which New P Ring enters the New P Register when an execute segment is entered (call, return, exchange, or intersegment branch instructions).

### LOCAL MEMORY

LM converts SVAs to RMAs using a Page Map RAM. It also retains the most recently used CM words in a Cache Memory. LM performs these two functions under micrand control. If the desired data exists in Cache Memory, the request to CM is disregarded.

### LM CONTROL

The following paragraphs describe LM Control shown on LM 1.0.

### Micrand Control

CST sends the LM micrand field to LM via SM. This field is part of the AC micrand which is described in section 3. The 11-bit LM field controls LM operations.

### CPU Requests

This part of LM Control analyzes CPU requests, determines when the current operation is completed, gates the requester's address and control information, and sends an accept back to the requester.

### Tags and Response

LM Control generates and sends an 8-bit tag to CMC for each request. This tag remains in CMC during the memory cycle, returns to LM with the data, and controls data sequences in LM (write) or OPI (read). The four CMC Tag formats (one for each function) are listed below.

<u>Function</u>	<u>Tag</u>
Cache Block Fill	0000XXYY
Cache Bypass	001WZZ—
Page Table Descriptor	010WP---
Stream	1WYYYYYY

XX = Tag File Number

YY = Word Number

W = Read/Write (Clear/Set)

ZZ = Destination Code

P = Even/Odd (Clear/Set)

### Cache Tag File

The Cache Tag File is a RAM which contains four entries in the format shown in figure 5-8. This information controls cache block fill operations, which occur when a requested word does not reside in Cache Memory. The requested word and three related words arrive from CM and replace the least recently used 4-word block. The other Cache Tag File locations are used if multiple block fill operations occur.

At the time a cache block is allocated, LM Control issues four requests to CM for the four words of the block. As the CM responses for these requests arrive at LM, the Cache Tag File controls the block fill operation for each word.

### SVA PATHS

The Address Mux (LM 1.0) selects from various inputs the SVA for distribution to the SVA to RMA Converter and to Cache Memory.

### SVA TO RMA CONVERSION

LM converts SVAs to RMAs (shown earlier in figure 5-1) using the Page Map RAMs (LM 1.0), which hold the most recently used Page Descriptors, or the System Page Table. LM also controls System Page Table operations (search, update).

#### Page Map RAMs

LM uses a set associative technique to convert an SVA to an RMA. Set associative means that four RAM sets are addressed simultaneously by the SPID. Each set contains a tag (SPID) and associated data (page frame address). Figure 5-7 shows the Page Map RAM formats. Each RAM contains up to 32 most recently used Page Descriptors from the system page table in CM. The lower five bits of the SPID (shifted page number) address the four Page Map RAM sets and the four Status RAM sets. The Hit Network compares the upper 33 bits of the SPID with the corresponding bits from each set at the addressed location. If they match and the Page Map RAM entry is valid, the upper Page Descriptor bits are sent to CMC as the page frame address (PFA) portion of the RMA. The SVA page offset bits are sent to CMC as the page offset portion of the RMA.

The misalignment of the valid (V) and parity (P) bits shown in figure 5-7 has no functional significance. These bits are arranged this way for physical reasons related to the RAM chips.

The two-bit least recently used (LRU) Code in the Status RAMs determines which set receives the new Page Descriptor from the system page table. The LRU entry has a code of 3; the most recently used (MRU) entry has a code of 0. The LRU entry is replaced by the new entry and becomes the MRU entry. Each of the other three sets increments their LRU code by one. This creates a new LRU entry at this address.



### Page Table Search Control

If the desired Page Descriptor is not in any Page Map RAM, the system page table is searched. This sequence is described earlier under SVA to RMA Conversion and is shown in figure 5-1. If the Page Descriptor is not found in the table, the page is not in CM. System software must then retrieve the page from an external storage device.

### CACHE MEMORY

Cache Memory (LM 1.0) is a high-speed buffer which contains up to 4,096 most recently used CM words. It reduces the effective CM access time and also acts as an instruction stack. Figure 5-8 shows the major components of Cache Memory.

Cache Memory contains two, three, or four associative sets. Each set holds 1,024 words organized in 256 4-word blocks. The lower 10 bits (51 through 60) of the SVA byte number address the four Cache Data RAM sets. Bits 51 through 58 address a block within the four Cache Tag, Cache Word Address Valid, Cache Tag File Address, and Cache Status RAM sets. Bits 59 and 60 select one of four words within the block. The Cache Tag Compare Network compares the SVA ASID and upper 18 bits of the byte number with the corresponding bits from each set of the Cache Tag RAM. If they match and the word is valid, the desired word is in Cache Memory. If not and this is a read operation, CM Read Data is fetched and enters this Cache Memory location from CMC at the same time it goes to other destinations within the CPU as LM Read Data. A later reference to that address reads the data directly from Cache Memory as LM Read Data. A CPU write operation changes the addressed CM location and the corresponding Cache Memory location (if valid).

### Cache Data RAM

The Cache Data RAM contains up to 1,024 4-word blocks of CM data. For a cache read operation, data from the hit set flows through the LM Read Data Mux to OPI during the second half of the major cycle. If no hit occurs, Allocation Control selects which set receives CM Read Data.

For a cache write operation in which a hit occurs, LM write Data from AC enters the hit set during the second half of the next major cycle. If no hit occurs during a write operation, nothing happens in Cache Memory.

### Cache Tag RAM

The Cache Tag RAM contains up to 1,024 entries in the format shown in figure 5-8. The SVA ASID and Byte Number are compared with Cache Address Register bits 16 through 31 and 33 through 50, respectively. A match between the upper SVA bits and a Cache Tag from one of the sets signifies a hit on that set. This allows data from the associated Cache Data RAM location to be used. The V bit indicates that this entry contains valid data. This bit sets during an allocate operation and clears during a purge or invalidate operation.

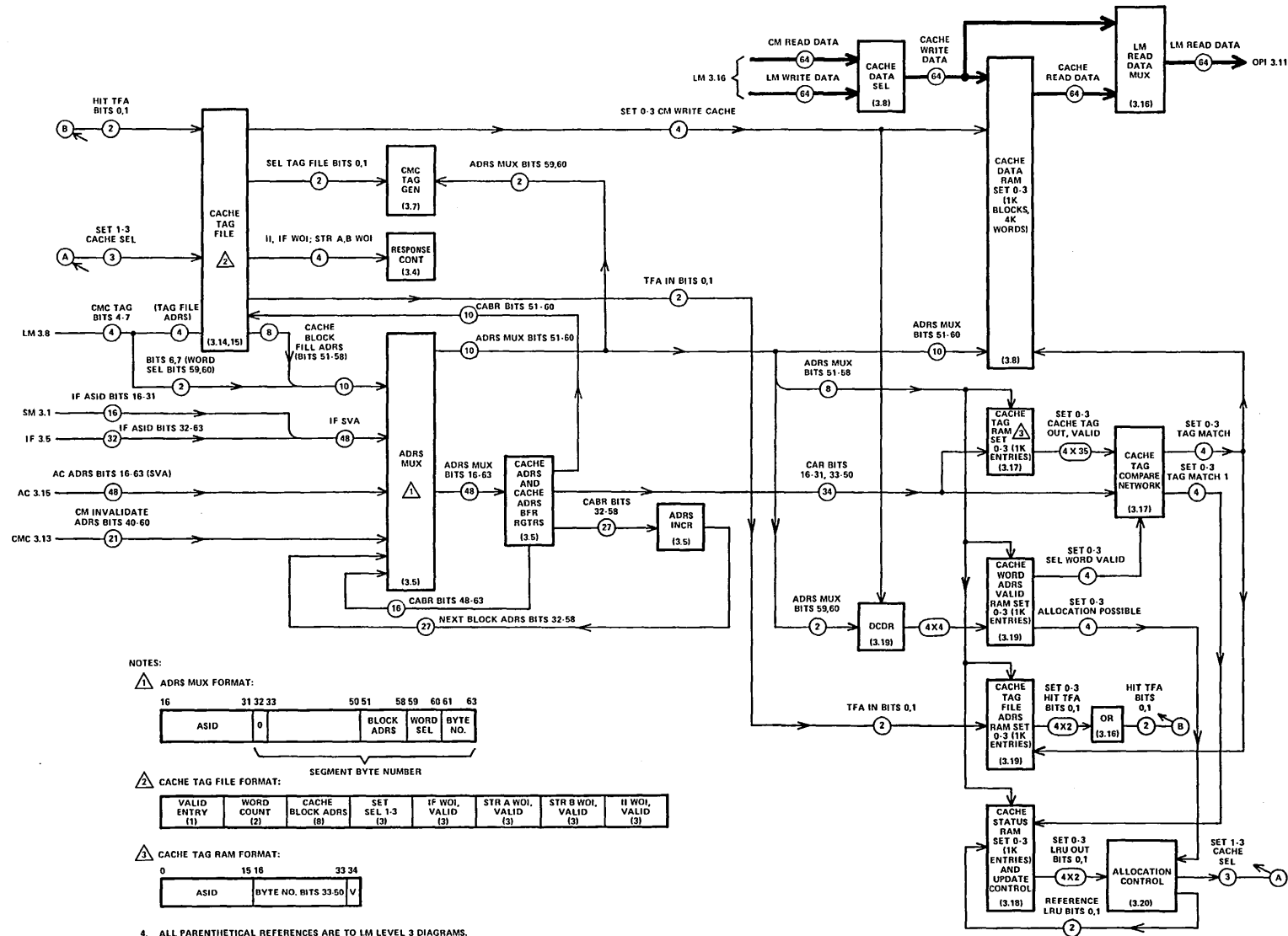


Figure 5-8. Cache Memory



#### Cache Word Address Valid RAM

The Cache Word Address Valid RAM contains up to 1,024 4-bit entries. This field identifies which Cache Memory word is valid in the associated block.

#### Cache Tag File Address RAM

The Cache Tag File Address RAM contains up to 1,024 2-bit entries. If a cache block is involved in a block fill operation, this RAM contains the address of the Cache Tag File which contains the block Fill information. This field is written at cache set allocate time. A CPU reference to a block involved in a block fill operation references this RAM. The CPU then places a word of interest (WOI) entry in the corresponding Cache Tag File.

#### Cache Status RAM

The Cache Status RAM contains up to 1,024 3-bit entries. The 2-bit LRU code is one of the conditions examined by Allocation and Update Control to determine which set to allocate. This field is written during an LRU update or an allocate. The Block Used bit is logically present on the AD112-C but it is not functional. In AD112-A/B the Block Used bit indicates a look-ahead request has been made to CM for the next sequential block. This bit sets during an allocate and clears when the look-ahead request is made.

#### Cache Parity Checker/Failing Chip Indicator

The Cache Parity Checker/Failing Chip Indicator has the dual function of detecting parity errors and locating failing cache memory chips. Specifically, it detects parity errors in the transmission of bits from the Tag RAM, Word Valid RAM, Status RAM, and Tag Field Address RAM to the Tag Parity Error Latch. These four RAMs are duplicated on four separate boards, creating four associative sets of cache memory. Four Tag Parity Error Latch sets hold the results of the parity check operations. The Parity Error Mux (4 sets) allows for the manual selection of Tag PE Latch bits 0-7 or 12-18 which are made accessible to test points for determining failing chips. Tag PE Latch bits 8-11 are directly accessible to test points.

#### Update Control

The Update Control updates the 2-bit LRU code in the Cache Status RAM whenever there is a cache reference. The set which was hit or allocated becomes the MRU and its LRU code is set to 0. Its former value is the Reference LRU. The other sets whose LRU is less than the Reference LRU increment by one. For example, assuming the following set LRU code and a set 2 hit:

<u>LRU Codes</u>		
<u>Set</u>	<u>Before Hit and Update</u>	<u>After Hit and Update</u>
0	00	01
1	01	10
2 (hit)	10	00
3	11	11

#### Allocation Control

The Allocation Control selects a set to be block filled when there is no hit on a cache reference. The selected set is the lowest numbered set that is enabled and not valid. If all enabled sets are valid, the set with the largest LRU code is selected.

## MODES OF OPERATION

The following paragraphs describe the various modes of operation for LM. Refer to LM 1.0.

### Cache Read

If a cache hit occurs, Cache Read Data goes from Cache Memory to the requester and no CM reference is needed. If a cache miss occurs, LM Control requests a 4-word block from CM. All four words enter the allocated set in Cache Memory. Only the word of interest goes to the requester.

### Cache Write

For CM write operations, Cache Memory is also written if a cache hit occurs. If a cache miss occurs, Cache Memory is unchanged.

### Cache Bypass Read or Write

Cache Memory is not used in this mode. LM performs the SVA to RMA conversion for the CM data reference.

### Real Memory Address

LM does not perform the SVA to RMA conversion and Cache Memory is not used. The CPU sends the RMA directly to CMC.

### Six-Byte Write

LM writes top of stack ring numbers into bytes 2 through 7 of exchange package words 38 through 51.

### Interrupt

Interrupt mode is the same as RMA write mode except the function code from AC is 1100 instead of 0000.

### Stream

Stream mode is used together with the RMA or cache bypass mode. It allows more than one outstanding CMC request. Exchange and BDP operations use this mode.

### Unconditional Look Ahead

Look Ahead mode is not used in Model AD112-C, but is functional in Models AD112-A/B. After a cache read hit, the Next Block Address points to the next sequential block in Cache Memory. If a cache hit occurs on the next block, nothing happens. If not, LM Control issues a CMC request for it.

### Conditional Look Ahead

Look Ahead mode is not used in Model AD112-C, but is functional in Models AD112-A/B. This is the same as the unconditional look ahead mode except conflicts are possible. If a CPU request conflicts with the look ahead request, the CPU request is granted and the look ahead request is aborted.

### Test

After the SVA to RMA conversion, the LM Read Data Mux sends the RMA to OPI. A page fault causes bit 32 to set. The page descriptor in the system page table is updated if necessary. Cache Memory is not used in this mode.

### Prevalidate

This mode is the same as test mode except RMA is not sent to OPI.

### Index

Page Table Search Control searches the entire table one entry at a time. After completing the search, the LM Read Data Mux sends the following information to OPI.

<u>Bits</u>	<u>Description</u>
1-28	RMA of Last Entry
34-39	Count of Entries Searched
63	Page Search Without Find

The page descriptors in the system page table are not updated. Cache Memory is not used in this mode.

### Purge

The purge mode invalidates all or part of the data in Cache Memory or Page Map. This operation is under micrand/instruction control.

### Invalidate

This mode invalidates the block in Cache Memory that is associated with a CM block that has been written by the IOU.

### Fake CM

This is a maintenance mode which allows the CPU to use Cache Memory when CM is not available. Cache Memory must be written under control of MAC (DEC bit 61). The allocated address may then be read back. If the request is to an address which is not in Cache Memory, the requester receives zeros. If the request is to an address for which data has not been loaded, the requester receives meaningless data.

## CONFLICTS

Many conflicts may occur in LM. The following paragraphs describe the major conflicts.

### Block Fill/Allocate

If a cache miss causes an allocate operation when a block fill is underway, the allocate operation is delayed one major cycle.

### Block Fill/Cache Write

If block fill and cache write operations conflict, the cache write operation is delayed one major cycle.

### Invalidate

An invalidate operation delays an allocate operation by two major cycles, or any other operation it interrupted by one major cycle.

### CM Busy

If CM is busy, all LM operations except cache hit reads are delayed until CM is finished.

### Write Followed by Read

If a cache hit read is to the same address as cache hit write was the previous cycle, the CM write reference is delayed one major cycle.

### Allocate/Word Not Valid

If an allocation is attempted to a cache address where all sets have at least one word not valid, the allocation is aborted. The CPU receives the requested word from CM.

### Read Access

If a CM word which is to be routed to the CPU arrives at LM during the same cycle as a cache hit read, the cache read is delayed one major cycle.

## LM SEQUENCES

The following paragraphs and the referenced level 3 diagrams describe typical LM sequences in which no conflicts occur. CPU is defined as any functional unit that can request a CM reference (ICP, IF, AC Stream A/C, and B).

#### LM Read Sequence

1. CPU sends request to Request Control (LM 3.0).
2. Accept Control (LM 3.0) sends accept to CPU.
3. Map Active FF (LM 3.1, 9) sets.
4. LM Active FF (LM 3.2) sets.
5. Address Mux (LM 3.5) sends SVA to SPID Register (LM 3.6) and Cache Address Register (LM 3.5).
6. Bypass and Real Address Selector (LM 3.1) determines this is not a cache bypass operation.
7. Cache Tag Compare Network (LM 3.17) determines if desired data resides in Cache Data RAM (LM 3.8). If tag match occurs, refer to steps 8 through 10. If not, resume sequence at step 11.
8. LM Read Data Mux (LM 3.16) sends data to CPU.
9. Response Translator (LM 3.4) sends response to CPU signifying that data is available.
10. LM Active FF (LM 3.2) clears.
11. (No Tag Match) Map Hit Network (LM 3.12) determines if desired page descriptor is in Page Map RAM (LM 3.12). If map hit occurs, proceed to step 12. If not, refer to Page Table Search Sequence and then return to step 12.
12. Tag File Valid Control (LM 3.14) determines that Cache Tag File (LM 3.14, 15) is not full. If it is full, sequence waits until current block fill operation completes.
13. Block fill FF (LM 3.20) sets to start block fill operation. Allocation Control (LM 3.20) allocates a cache set. CMC Tag selects Cache Tag File (LM 3.14, 15).
14. Cache Tag File (LM 3.14, 15) loads with set select code (LM 3.20), cache address (LM 3.14), and the requester's word of interest (WOI) within the cache block (LM 3.15).
15. CMC Request Control (LM 3.7, 9) sends LM Request to CMC for first word of block fill.
16. Real Memory Address Network (LM 3.10) forms RMA for first word of block fill. If map hit occurs, proceed to step 17. If not, refer to Page Table Search Sequence and then return to step 17.
17. Address Incrementer (LM 3.10) increments RMA.
18. LM Active FF (LM 3.2) resets.
19. LM Counter (LM 3.3), used to terminate cache block fill at an appropriate count, resets to zero.
20. CMC Request Control (LM 3.7, 9) sends LM Request to CMC for second word of block fill.
21. Address Incrementer (LM 3.10) increments RMA.
22. LM Counter (LM 3.3) increments to one.

23. CMC Request Control (LM 3.7, 9) sends LM Request to CMC for third word of block fill.
24. Address Incrementer (LM 3.10) increments RMA.
25. LM Counter (LM 3.3) increments to two.
26. CMC Request Control (LM 3.7, 9) sends LM Request to CMC for fourth word of block fill.
27. Address Incrementer (LM 3.10) increments RMA.
28. CMC Tag Generator (LM 3.7) generates and sends CMC Tag Bits 0 through 7 to CMC. Bits 4 and 5 define tag file for this block fill. Bits 6 and 7 specify word within block being requested.
29. Map Active FF (LM 3.1, 9) clears.
30. Block Fill FF (LM 3.20) clears.
31. CMC sends CMC Response to CMC Response Decoder (LM 3.8) signifying that first block fill word is available. CMC sends CMC response and Tag one major cycle prior to data.
32. Tag File Valid Control (LM 3.14) selects a Cache Tag File (LM 3.14, 15) location according to returned CMC Tag bits 4 and 5.
33. Address Mux (LM 3.5) sends Cache Block Fill Address to Cache Data RAM (LM 3.8). Cache block address and set select are from selected Cache Tag File (LM 3.14). Word number is from CMC Tag bits 6 and 7.
34. Cache Data Selector (LM 3.8) gates first block fill word from CMC to Cache Data RAM (LM 3.8).
35. Decoded word number (CMC Tag bits 6, 7) sets appropriate word valid bit in Word Address Valid RAM (LM 3.19).
36. Word of Interest Check (LM 3.15) determines if this block fill word is word needed by CPU requester. If it is, proceed to step 37. If not, to step 39.
37. Response Translator (LM 3.4) sends response to CPU requester.
38. LM Read Data Mux (LM 3.16) sends data to CPU.
39. Repeat steps 31 and 33 through 38 for second, third, and fourth block fill words.

#### Cache Bypass LM Read Sequence

1. CPU sends request to Request Control (LM 3.0).
2. Accept Control (LM 3.0) sends accept to CPU.
3. Map Active FF (LM 3.1, 9) sets.
4. LM Active FF (LM 3.2) sets.
5. Address Mux (LM 3.5) sends SVA to SPID Register (LM 3.6) and Cache Address Register (LM 3.5).

6. Bypass and Real Address Selector (LM 3.1) determines this is a cache bypass operation.
7. Map Hit Network (LM 3.12) determines if desired page descriptor is in Page Map RAM (LM 3.12). If map hit occurs, proceed to step 8. If not, refer to Page Table Search Sequence and then return to step 8.
8. Map Enable Control (LM 3.9) determines if CM port is busy. If busy, sequence waits until CM is not busy. If not busy, proceed to step 9.
9. CMC Request Control (LM 3.7, 9) sends LM Request to CMC.
10. CMC Tag Generator (LM 3.7) generates and sends CMC Tag to CMC. A portion of tag defines CPU requester.
11. Real Memory Address Network (LM 3.10) forms RMA.
12. CMC Function Code Control (LM 3.7) sends CMC Function Code to CMC.
13. Map Active FF (LM 3.1, 9) clears.
14. CMC and CM cycle at appropriate memory location.
15. One major cycle prior to sending data, CMC sends CMC Tag and Response to CMC Tag Decoder (LM 3.8).
16. LM Read Data Mux (LM 3.16) sends data to CPU.
17. Response Translator (LM 3.4), under control of returning CMC Tag, sends response to CPU requester signifying that data is available.
18. LM Active FF (LM 3.2) clears.

#### LM Write Sequence

1. CPU sends request to Request Control (LM 3.0).
2. Accept Control (LM 3.0) sends accept to CPU.
3. Map Active FF (LM 3.1, 9) sets.
4. LM Active FF (LM 3.2) sets.
5. Address Mux (LM 3.5) sends SVA to SPID Register (LM 3.6) and Cache Address Register (LM 3.5).
6. Map Hit Network (LM 3.12) determines if desired page descriptor is in Page Map RAM (LM 3.12) and if Page Modified bit is set. If map hit occurs and bit is set, proceed to step 7. If not, refer to Page Table Search Sequence and then return to step 7.
7. Cache Tag Compare Network (LM 3.17) determines if data to be written is in Cache Data RAM (LM 3.8). If tag match occurs and this is not a cache bypass operation, LM Write Data enters Cache Data RAM.

Map Enable Control (LM 3.9) determines if CM port is busy. If busy, sequence waits until CM is not busy. If not busy proceed to step 8.

8. CMC Request Control (LM 3.7, 9) sends LM Request to CMC.
9. Real Memory Address Network (LM 3.10) forms RMA.
10. CPU sends LM Write data (LM 3.16) to CMC (through AC).
11. LM Active FF (LM 3.2) clears.
12. Map Active FF (LM 3.1, 9) clears.
13. After completion of write sequence in CMC, Response Translator (LM 3.4) sends Response to CPU requester under control of returning CMC Tag and Response (LM 3.8). If CMC Error Response Decoder (LM 3.8) detects uncorrected error within CM or CMC (specified by CMC Response Code), the decoder sends CM Detected Malfunction During Write to ICC.

#### Page Table Search Sequence

1. Map Hit Network (LM 3.12) determines that desired page descriptor is not in Page Map RAM (LM 3.12).
2. Map Enable Control (LM 3.9) determines if CM port is busy. If busy, sequence waits until CM is not busy. If not busy, proceed to step 3.
3. CMC Request Control (LM 3.7, 9) sends LM Request to CMC for initial page descriptor in system page table.
4. Page Table Address (PTA) Register (LM 3.10) and output of Hash Network (LM 3.6) (which combines ASID and page number) are fed to RMA Mux (LM 3.10), which formats initial address (always on even word boundary) into system page table. CM Address/Data Transmitter (LM 3.16) sends initial address to CMC.
5. LM Counter (LM 3.3), which counts number of page table requests, increments to one.
6. Even Request Outstanding FF (LM 3.11) sets.
7. Request Odd FF (LM 3.11) sets.
8. CMC Request Control (LM 3.7, 9) sends LM Request to CMC for second page descriptor.
9. Real Memory Address Network (LM 3.10) forms RMA for second page descriptor by selecting previous address through Address Incrementer (LM 3.10) and incrementing by one.
10. Odd Request Outstanding FF (LM 3.11) sets.
11. LM Counter (LM 3.3) increments to two.
12. CMC sends CMC Tag and Response to CMC Tag Decoder (LM 3.8) one major cycle prior to sending data.
13. Page Table Hit Control (LM 3.11) determines if desired page descriptor is at first location in system page table by comparing SPID portion of returning CMC word with LM SPID Register (LM 3.6). If page table hit occurs, proceed to step 14. If not, resume the sequence at step 24.



14. Page Map Allocation Control (LM 3.13), using a method similar to Cache Allocation Control (LM 3.20), enables writing initial page descriptor into Page Map RAM (LM 3.12).
15. Page Table Search Control (LM 3.11) determines if CM Read Data Bits 2 and 3 (Page Descriptor Used and Modified) are both set (LM write) or bit 2 is set (LM read). If true, return to LM Read Sequence step 12 or 17, Cache Bypass LM Read Sequence step 8, or LM Write Sequence step 7. If not true, proceed to step 16 for page table update sequence.
16. Page Table Hit Address Check (LM 3.11) examines CMC Tag bit 4 to determine if page table hit occurred on an even address in system page table. If it did (bit 4 clear), Address incrementer (LM 3.10) decrements RMA. If not, proceed to step 17.
17. Map Enable Control (LM 3.9) determines if CM port is busy. If busy, sequence waits until CM is not busy. If not busy, proceed to step 18.
18. Page Table Search Control (LM 3.11) generates Set Lock Function signal.
19. CMC Request Control (LM 3.7, 9) sends LM Request to CMC for page table update operation.
20. Real Memory Address Network (LM 3.10) forms RMA for page descriptor to be updated.
21. Page Table Request FF (LM 3.16) sets.
22. Address/Data Mux (LM 3.16) sends LM Write Data Bits 0 through 7 (page table update data) to CMC.
23. Return to LM Read Sequence step 12 or 17, Cache Bypass LM Read Sequence step 8, or LM Write Sequence step 7.
24. Page Table Search Control (LM 3.11) and Page Fault Test (LM 3.13) determine if Continue (CM Read Data Bit 1) is set in page descriptor and LM Counter (LM 3.3) equals less than 32. If so, proceed to step 25. If not, resume sequence at step 30.
25. Map Enable Control (LM 3.9) determines if CM port is busy. If busy, sequence waits until CM is not busy. If not busy, proceed to step 26.
26. CMC Request Control (LM 3.7, 9) sends LM Request to CMC for next page descriptor in system page table.
27. Real Memory Address Network (LM 3.10) forms RMA for next page descriptor by incrementing previous address by one.
28. LM Counter (LM 3.3) increments.
29. Repeat steps 6 through 24 until page table hit occurs, Continue is clear, or LM Counter (LM 3.3) equals 32. If Page Fault Test (LM 3.13) determines that Continue is clear or LM Counter equals 32 without a page table hit, proceed to step 30.
30. Response Translator (LM 3.4) sends response to CPU requester.
31. Page Fault Detector (LM 3.13) generates Page Search Without Find signal.
32. LM Active FF (LM 3.2) clears.

33. Map Active FF (LM 3.1, 9) clears.

34. Page Search Without Find signal (step 31) causes an interrupt which allows system software to locate page, place page in CM, update system page table, and resume interrupted sequence.

#### Cache Hit and LRU Update Sequences

Figures 5-9 and 5-10 show these sequences for LM read and write operations.

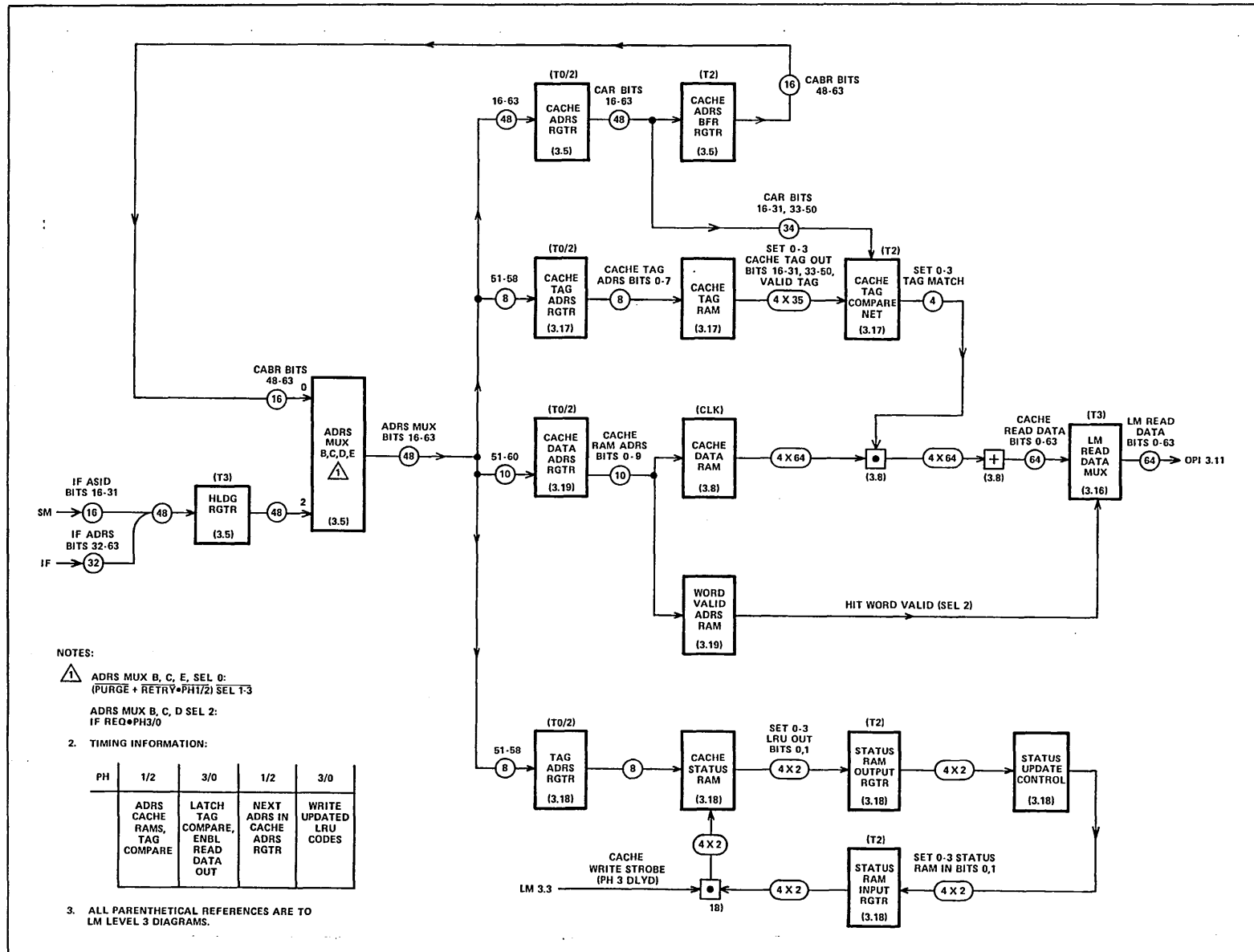


Figure 5-9. LM Read, Cache, Hit, LRU Update

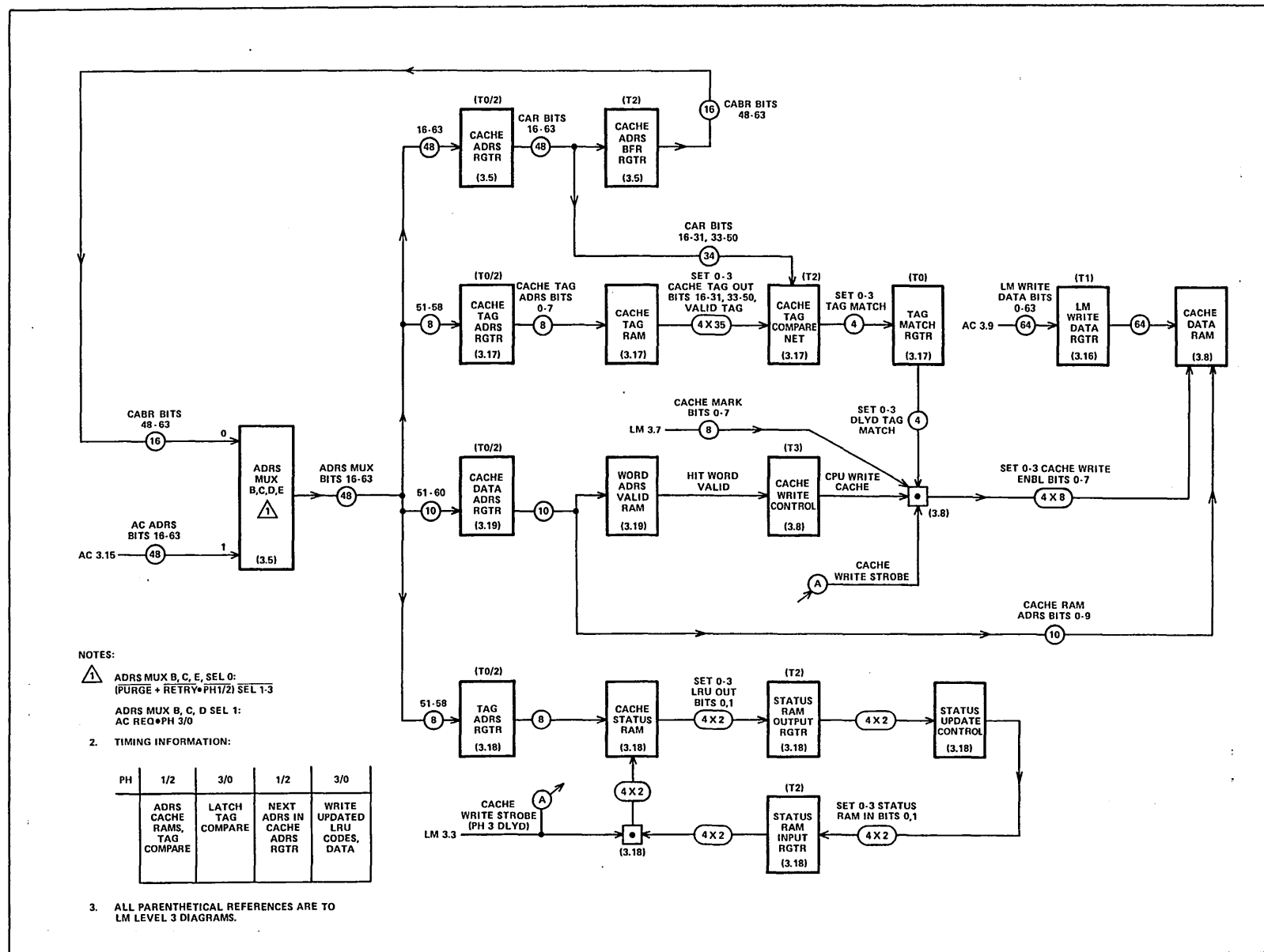


Figure 5-10. LM Write, Cache Hit, LRU Update

## CENTRAL MEMORY CONTROL

CMC provides access to Central Memory (CM). CMC receives requests from four sources. LM (A), LM (B), Input/Output Unit (IOU), and an Auxiliary (Aux) port all have access to CM through CMC. CMC resolves CM bank conflicts and simultaneous CM request conflicts.

### INPUT PORT INTERFACE SIGNALS

Each requesting port sends address, data, and control to CMC. These signals are described in the following paragraphs. Refer to CMC 1.0.

#### Write Data

This 64-bit (plus 8 parity bits) path contains the data to be stored in CM during a write operation. A parity bit accompanies each data byte. For LM (A) and LM (B) ports, the data is time-multiplexed on the same lines as the address.

#### Address

This 27-bit or 29-bit (plus 4 parity bits) path contains the word address that is to be accessed in CM. The address from IOU and Aux ports is 27 bits, and from LM (A) and LM (B) is 29 bits. Only the lower-order 24 bits are used to address CM. (For Model AD112-A/B only the lower-order 21 bits are used.) The upper bit (bit 0) is the AC Address Adder sign bit. If bit 0 is set, an address specification error has occurred. Bits 1-4 are not used. (For Model AD112-A/B bits 1-7 are not used.) For LM (A) and LM (B) ports, the address is time-multiplexed on the same lines as the data.

#### CMC Mark

This 8-bit (plus parity bit) path contains a mark bit for each byte in the Write Data word. The mark bits determine which bytes are to be written. Only bytes with accompanying mark bits are written into CM.

#### CMC Tag

This 8-bit (plus parity bit) path contains information from the requesting port, which uses it to route returning data and responses. This tag returns to the requesting port unchanged with CMC Response. It is sent one major cycle prior to CM Read Data and CMC Response Code through an Output Port. Except for exchange and IOU write requests, CMC does not use this tag, but simply delays it during the CM reference.

### CMC Function Code

This 4-bit (plus parity bit) path contains the type of CM operation desired by the requesting port. CMC translates the Function Code as follows:

<u>Code</u>	<u>Function</u>
0	Read
2	Write
4	Read and Set Lock
5	Read and Clear Lock
6	Exchange
A	Read Free Running Counter (LM ports only)
B	Refresh Counter Resync
C	Interrupt

These functions are described later. All other codes are illegal and result in a reject response. Code A also results in a reject response if sent from an IOU or Aux port.

### Request

This signal initiates CMC and enables all of the preceding signals.

### OUTPUT PORT INTERFACE SIGNALS

Each requesting port receives data and control from CMC. These signals are described in the following paragraphs. Refer to CMC 1.0.

### CM Read Data

This 64-bit (plus 8 parity bits) path contains the data returned from CM during a read operation. A parity bit accompanies each data byte.

### CMC Tag

This 8-bit (plus parity bit) path contains a delayed copy of the CMC Tag sent to the Input Port after a read or write Request. The CMC Tag returns to the requesting port one major cycle prior to the corresponding CM Read Data and CMC Response Code.

### CMC Response

This signal informs the requesting port that the desired CM Read Data will be available in one major cycle. CMC Response enables CMC Tag and occurs one major cycle prior to the corresponding CM Read Data and CMC Response Code.

### CMC Response Code

This 3-bit (plus parity bit) path contains the type of response being returned to the requesting port. CMC encodes the responses according to the function being performed as follows:

<u>Code</u>	<u>Response Type</u>
0	Write Response
1	Write Response Uncorrectable Error
2	Write Response Corrected Error
3	Interrupt Response
4	Read Response
5	Read Response Uncorrectable Error
6	Read Response Corrected Error
7	Reject

### Interrupt

This signal causes an interrupt to be sent to the port specified by Write Data bits 60 through 63 as follows:

<u>Bit</u>	<u>Port</u>
60	AUX
61	LM (B)
62	IOU
63	LM (A)

### Port Busy

This signal stops the flow of requests when an Input Port is unable to accept additional requests due to a conflict. The IOU and Aux ports each contain request buffers which allow eight requests to be stored before Port Busy is sent.

## FUNCTIONS

CMC performs one of eight functions depending on the CMC Function Code from the requesting port. These functions are listed earlier under CMC Function Code and described in the following paragraphs. Refer to CMC 1.0.

### Read

CMC Function Code 0 initiates this operation. It causes Read Data from CM to be sent to the requesting port through the SECDED network and an Output Port. CMC Response Code and Tag precede the CM Read Data to the requesting port.

## Write

CMC Function Code 2 initiates this operation. It causes Write Data from the requesting port to be sent to CM through an Input Port and the Write ECC Generator. CMC Control generates CM Write to enable Write Data into CM. If one or more Mark bits are clear, CMC Control generates Two Pass Function which causes a read/modify/write (partial write) operation to occur in CM. The modifying Write Data enters CMC through an Input Port, goes through the Write ECC Generator and loads into a holding register in CM. Write Data from that holding register and Read Data from a CM bank (by way of SECDED) enter the Partial Write Network and ECC Generator in CMC. The combination of the two words creates a PW Data word, which returns to CM after an ECC is generated. Mark bits control the Partial Write Network. CM word bytes which have corresponding mark bits set are modified. The remaining bytes return to CM unchanged. The Output Port sends CMC Response Code and Tag to the requesting port at the end of the write operation.

## Read and Set Lock; Read and Clear Lock; Exchange

CMC Function Codes 4, 5, and 6 initiate these operations. They cause data in CM to be modified according to the Mark bits and Write Data from the requesting port. These are partial write functions which enable the CM Write and Two Pass Function signals. The resulting Partial Write Data is rewritten in CM. CMC Response Code and Tag precede the unmodified CM Read Data to the requesting port.

### Read and Set Lock

The Partial Write Network forms a logical OR of the marked Write Data and the corresponding Read Data bytes from CM.

### Read and Clear Lock

The Partial Write Network forms a logical AND of the marked Write Data and the corresponding Read Data bytes from CM.

### Exchange

The Partial Write Network exchanges the marked Write Data with the corresponding Read Data bytes from CM.

## Read Free Running Counter

CMC Function Code A initiates this operation. It allows LM (A) or LM (B) to read Maintenance Register B0 which contains the Free Running Counter. CMC Response Code and Tag precede the CM Read Data to LM. The counter fills the rightmost 48 bit positions. Zeros fill the leftmost 16 bit positions.



### Refresh Counter Resync

CMC Function Code B initiates this operation. It forces a conflict on the requesting port which stops the next request from being honored. The forced conflict clears when a bank 0 refresh cycle occurs. This allows the following requests to be processed normally and in sync with refresh. The Output Port sends CMC Response Code and Tag to the requesting port at the end of refresh counter resync operation.

### Interrupt

CMC Function Code C initiates this operation. It causes one or more ports to be interrupted (including the requester's own) based on Write Data bits 60 (AUX), 61 (LM (B)), 62 (IOU), and 63 (LM (A)).

## MAJOR COMPONENTS

The following paragraphs describe the CMC major components shown in CMC 1.0.

### Input Ports

The Input Ports receive address, data, and control signals from the requesting ports or an address from the Refresh Address Generator. Conflict and Bank Busy Control determines which port has priority.

### Aux and IOU Buffer Memories

The Aux and IOU ports each have 16-word Buffer Memories. If a request cannot be honored due to a conflict, the request (and additional requests) is stored until the conflict is resolved. When eight requests have been stored, Port Busy activates to block additional requests.

### Write Error Correction Code (ECC) Generator

The Input Ports send write data and eight parity bits to the Write ECC Generator. If SECDED mode is selected, eight ECC bits are generated and accompany the Write Data to CM. Eight parity bits are also sent to CM to be stored with the Write Data for two pass functions. If parity mode is selected, the Write Data and eight parity bits pass through to CM.

### Conflict and Bank Busy Control

This network receives the requests, resolves port and bank conflicts, and generates Go Banks 0 through 7 Enable signals. Address, data, and control signals associated with the request are gated through the Input Port when that request has priority and the requested bank is not busy. Port requests are processed on a time slice and last used basis. This network also services Refresh Requests which have priority over all other requests.

### Refresh Address Generator

The Refresh Address produces a periodic read cycle on every CM location. Refresh Request occurs every 15.25 major cycles and refreshes large blocks of CM. (Block size is dependent on CM size.) All locations are refreshed within 4 milliseconds.

### CMC Control and Delay

This network delays address and control signals from near the beginning of a CM cycle to near the end. This delay network provides proper timing and control of the various functions.

### Single Error Correction/Double Error Detection (SECDED)

The SECDED network corrects single-bit errors and detects multiple-bit errors. This network regenerates the ECC for every word read from CM. It compares this ECC with the ECC in the word from CM. If they are different, an error occurred and is logged in one of the Maintenance Registers.

### Partial Write Network

The write, read and set lock, read and clear lock, and exchange functions described earlier use the Partial Write Network. All of these functions require two CM cycles. This network combines CM Write and Read Data according to the intended function and the status of the Mark bits.

### Partial Write ECC Generator

This network is the same as the Write ECC Generator described earlier. It regenerates the ECC code for Partial Write Data entering CM.

### Output Ports

The Output Ports send data and control signals to the requesting ports. The data source is determined by a Read Select code generated in CMC Control when the request was initially made and delayed until needed. CM Read Data originates from CM or Free Running Counter for LM ports. For the IOU and AUX ports, CM Read Data originates only from CM.

### Maintenance Registers

CMC contains the nine maintenance registers listed below.

<u>Register</u>	<u>Function</u>
00	Status Summary
10	Element Identifier
12	Degrade Status
20	Environment Control
21	Bounds Register
A0	Corrected Error Log
A4	Uncorrectable Error Log 1
A8	Uncorrectable Error Log 2
B0	Free Running Counter

MAC can write all of the registers except 00, 10, and 12 which are preset by removable connectors. MAC can access all of the registers except B0. On a write operation, MAC sends a Write signal, a Maintenance Register Select, and a Maintenance Register Byte Count for each of eight bytes of MAC Data. The absence of the Write signal causes the byte to be read and sent to MAC. Refer to section 2 which describes MAC reads and writes of the Maintenance Registers. Also, refer to CMC 3.13 and 3.14 which depict the Maintenance Registers.

### Free Running Counter

This is a 1-microsecond counter which is contained in Maintenance Register B0. It can be written by MAC and can be read only by LM (A) or LM (B).

### Bounds Fault Detector

This network compares the current Address (modulo 512; only bits 8 through 19 are compared) with the contents of Maintenance Register 21 (Bounds Register). If the current address is out of bounds and bounds checking is enabled, a bounds fault is reported to Maintenance Register A4.

### TIMING

Figures 5-11 and 5-12 show the CMC timing from request in to CM read data out for each port. Figures 5-13 and 5-14 show the CMC timing for a partial write request from LM (A).

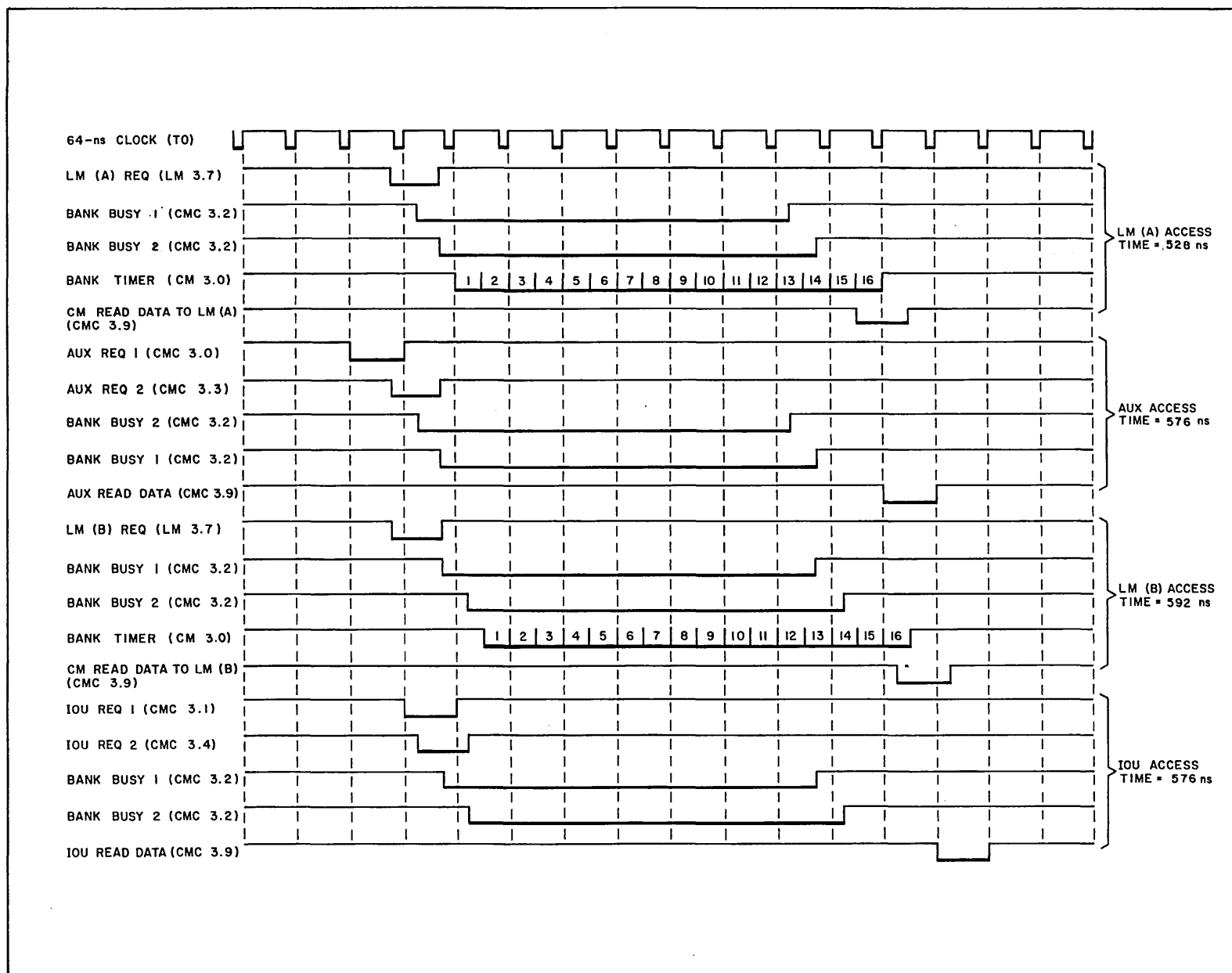


Figure 5-11. CMC Request/CM Read Data Out Timing (AD112-C)

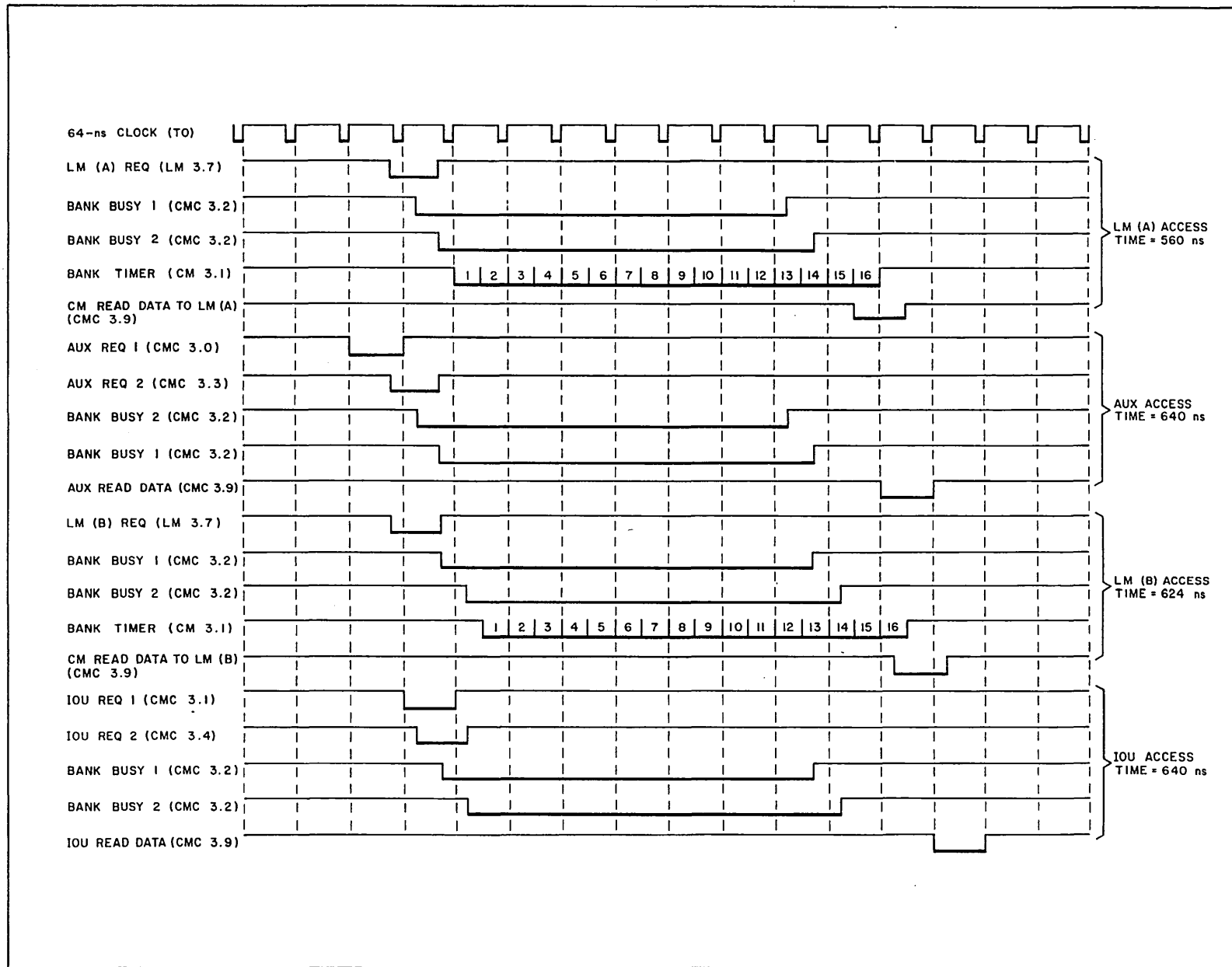


Figure 5-12. CMC Request/CM Read Data Out Timing (AD112-A/B).

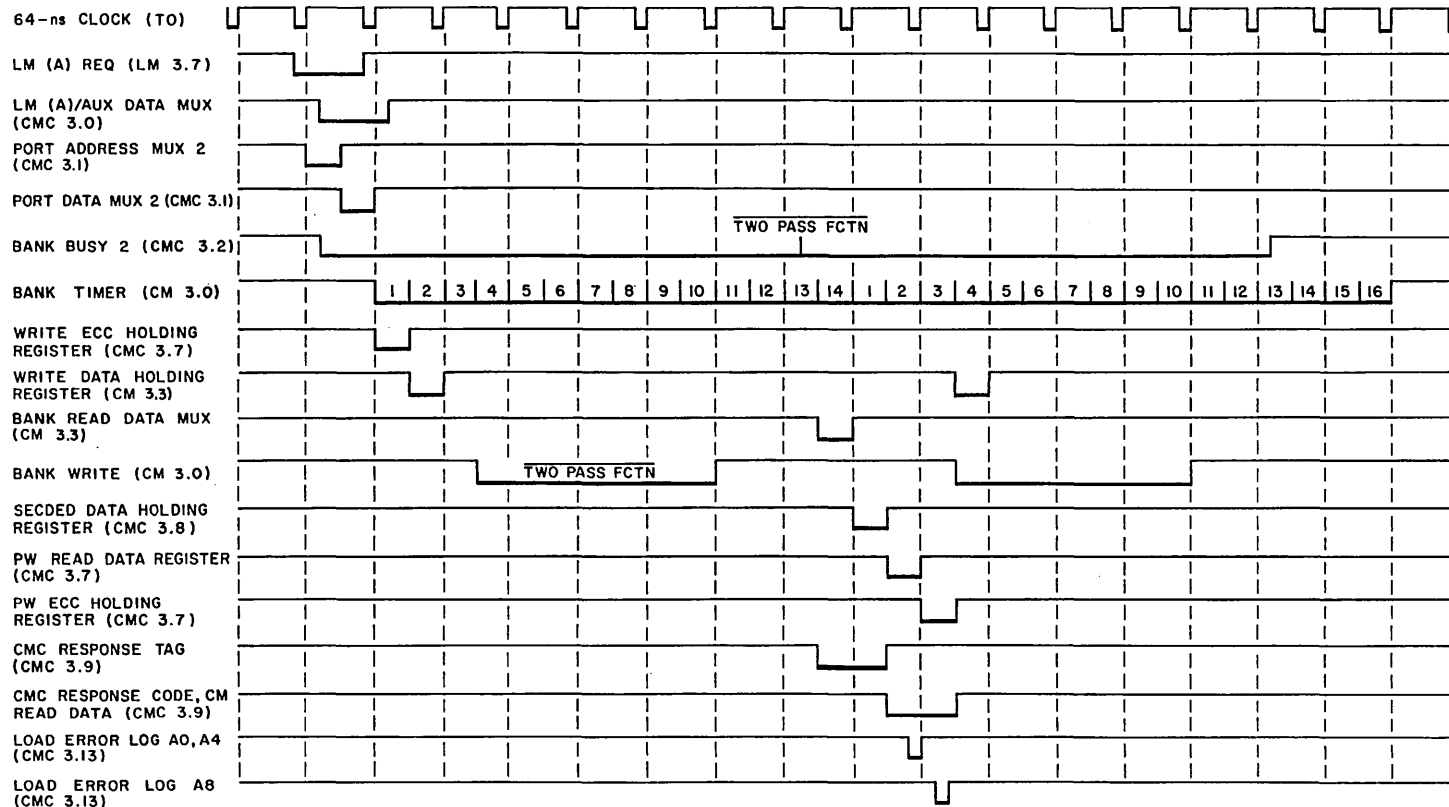


Figure 5-13. CMC Timing (AD112-C)

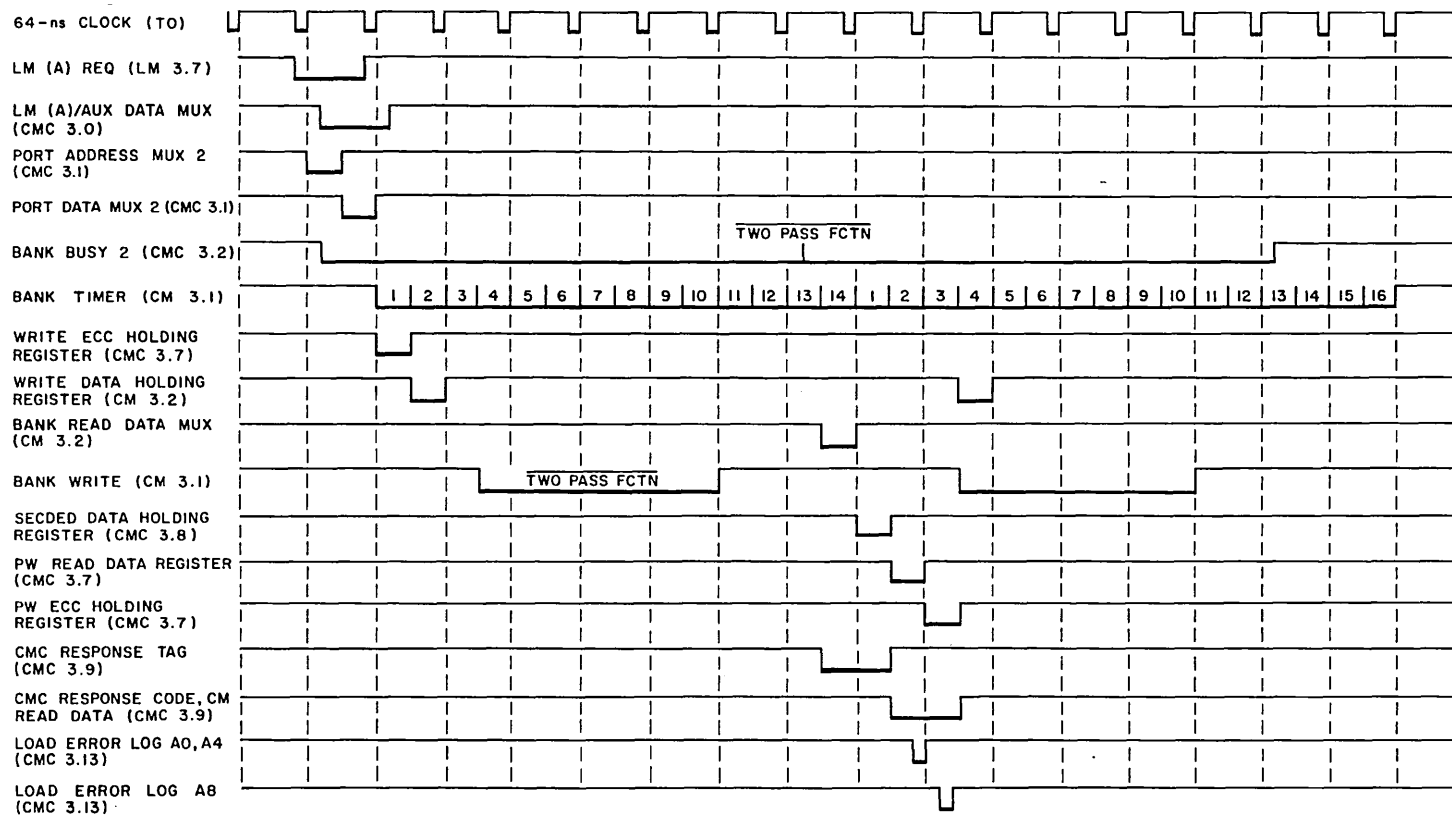


Figure 5-14. CMC Timing (AD112-A/B)

## CENTRAL MEMORY (BS137-A)

### NOTE

The following description applies only to the BS137-A central memory (CM). Refer to Central Memory (BS213-A, BS214-A, BS215-A) for a description of these CMs.

CM provides storage for LM (A), LM (B), IOU, and the Aux port. These ports access CM through CMC. Eight Memory Banks store 524K, 1049K, 1573K, or 2097K, 64-bit (plus 8 code bits) words depending on the configuration. CM performs read, write, and read/modify/write (partial write) operations.

### INTERFACE SIGNALS FROM CMC

CMC sends address, data, and control to CM. These signals are described in the following paragraphs. Refer to CM 1.0.

#### Address

This 21-bit (plus 3 parity bits) path contains the address in memory which is to be accessed. The address format is shown in CM 1.0. Each address byte is accompanied by a parity bit. Any CMC input parity error inhibits writing into the addressed location.

#### Go Bank

This signal initiates one of eight banks for a reference.

#### Refresh Request

This signal and Go Bank indicates the current reference is for a refresh cycle in the selected bank. Refresh forces all quadrants to cycle the accompanying refresh address.

#### Refresh Request or Interleaved Mode

This signal places CM in the normal mode of operation, where sequential addresses result in phased-bank operation. Absence of this signal places CM in the noninterleaved mode for maintenance purposes, where sequential addresses are located in the same bank.

#### CM Write

This signal indicates that the current CM request is for a write operation.



#### Two Pass Function

This signal indicates that the current CM request is for a partial write operation requiring two bank cycles.

#### Multiple Bit Error

This signal indicates that the SECDED network in CMC detected a multiple bit error in the data read from CM. If this is a partial write operation, Write and Second Pass Control inhibits writing the modified data during the second half of the operation.

#### Enable Long Cycle

This signal delays Clear Bank 0 Busy after the second half of a partial write operation involving the Free Running Counter (FRC) in CMC. Bank 0 Busy forces a conflict on any port which requests an FRC read. The conflict ensures that a subsequent FRC read request by the same port or any other port results in a different count.

#### Partial Write Data/Code

This 64-bit (plus 8 error correction code bits) path contains the data to be stored during the second half of a partial write operation.

#### Write Data/Parity, Code

This 64-bit (plus 8 parity bits and 8 error correction code bits) path contains the data to be stored during a write operation. The eight parity bits are not stored, but are returned to CMC along with the Write Data during a partial write operation.

#### INTERFACE SIGNALS TO CMC

CM sends data and control to CMC. These signals are described in the following paragraphs. Refer to CM 1.0.

#### Clear Bank Busy

This signal indicates that a bank cycle is nearly complete. It allows CMC to generate another Go Bank for this bank.

#### Read Data/Code

This 64-bit (plus 8 error correction code bits) path contains the data retrieved from a Memory Bank during a read operation.

### Write Data/Parity

This 64-bit (plus 8 parity bits) path contains a copy of the Write Data sent to the Partial Write/Write Data Mux during a partial write operation. This data goes to the Partial Write Network in CMC.

### MEMORY ORGANIZATION

CM 2.0 shows the memory organization. Memory consists of standard quadrant 0 and optional quadrants 1, 2, and 3. Each quadrant stores 524,288 64-bit (plus 8 parity or error correction code bits) words in eight banks. Each quadrant of each bank is located on two array boards. Each board contains 144 memory chips arranged in four logical rows of 36 chips each. Each memory chip is a 16,384 by 1-bit RAM array. Therefore, a logical row stores 16,384 words, a bank stores up to 65,536 words, a quadrant stores up to 524,288 words, and memory stores up to 2,097,152 words.

### MEMORY ADDRESSING

Go Bank selects one of eight banks. Quadrant Select selects two array boards within the bank. Row Select selects one of four logical rows of chips on the array boards. Row/Column Address selects one address within each memory chip in the logical row. Refer to CM 1.0.

### FUNCTIONS

CM performs one of four functions depending on the condition of Go Bank, CM Write, and Two Pass Function signals from CMC. A 32-nanosecond clock controls timing. These functions are described in the following paragraphs. Refer to CM 1.0.

#### Read

1. Go Bank initiates a read function when CM Write and Two Pass Function are absent. Go Bank starts Bank X Timing Control which sequences selected bank through addressing and read cycles.
2. Bank Control sends Clear Bank Busy to CMC after 11 clock periods (352 nanoseconds).
3. Bank Read Data Mux sends Read Data/Code from the addressed Memory Bank to CMC after 13 clock periods (416 nanoseconds).

#### Write

1. Go Bank and CM Write initiate a write function when Two Pass Function is absent. Go Bank starts Bank X Timing Control which sequences selected bank through addressing and write cycle.
2. The addressed Memory Bank stores Write Data/Code from CMC after 3 clock periods (96 nanoseconds) if no address parity error is detected.
3. Bank Control sends Clear Bank Busy to CMC after 11 clock periods (352 nanoseconds).

### Partial Write

1. Go Bank, CM Write, and Two Pass Function initiate a partial write function. Go Bank starts Bank X Timing Control which sequences selected bank through addressing and partial write cycle. Bank X timing Control makes two passes.
2. Bank Read Data Mux sends Read Data/Code from the addressed Memory Bank to CMC after 13 clock periods (416 nanoseconds).
3. Bank Write Data Mux sends Write Data/Parity from CMC back to CMC after 14 clock periods (448 nanoseconds).
4. The addressed Memory Bank stores Partial Write Data/Code from CMC after 17 clock periods (544 nanoseconds).
5. Bank Control sends Clear Bank Busy to CMC after 25 clock periods (800 nanoseconds.)

### NOTE

If an address parity error occurs, the partial write function becomes a read cycle. If a Multiple Bit Error occurs during the first half of the partial write function, the second half becomes a read cycle.

### Long Read Cycle

This function is required for the Read Free Running Counter function described earlier for CMC. Data is read from Memory Bank 0 but is not used. This sequence creates a conflict if another port requests Memory Bank 0 during the Read Free Running Counter operation.

1. Go Bank 0 and Two Pass Function initiate a long read cycle when CM Write is absent. Go Bank 0 starts Bank 0 Timing Control which sequences bank 0 through address and long read cycle. Bank 0 Timing Control makes two passes.
2. Bank Read Data Mux sends Read Data/Code from Memory Bank 0 to CMC after 13 clock periods (416 nanoseconds).
3. CMC sends Enable Long Cycle to Clear Bank Busy Control.
4. Bank Read Data Mux sends Read Data/Code from Memory Bank 0 to CMC after 27 clock periods (864 nanoseconds).
5. Bank Control sends Clear Bank 0 Busy to CMC after 29 clock periods (928 nanoseconds). This creates a conflict.

## PARTIAL WRITE OPERATION

Several CM and CMC level 3 diagrams depict the hardware needed to perform the partial write operation. Figure 5-15 shows these circuits and includes a reference to the corresponding level 3 diagrams. Figure 5-16 shows CM bank timing for this operation and also for the read and write operations.

The partial write operation requires two memory references (passes). During the first pass, a word is read from a Memory Bank address and sent to the Partial Write Network along with a word from LM. The Partial Write Network produces a Partial Write Data word by replacing or modifying one or more Read Data bytes with marked Write Data bytes. During the second pass, this Partial Write Data word is written into the same Memory Bank address. The following steps describe the sequence of events.

1. CMC determines that a partial write operation is to be performed in a CM bank and generates following control signals.
  - Bank Busy 2 (not shown on figure 5-15).
  - CM Write.
  - Two Pass Function.
  - Mark (one or more of eight).
  - Partial Write or Exchange, Read and Set Lock, Read and Clear Lock (one of three depending on type of partial write operation).
  - Go Bank Enable.
2. Bank Timer starts counting for first pass.
3. LM Address passes through Input Port and enters holding registers in Address and Write Control.
4. Write and Second Pass Control generates Enable Data Latch.
5. LM Write Data passes through Input Port, Write ECC Generator, and enters holding register in Partial Write/Write Data Mux.
6. Address and Write Control sends Row Address to Memory Bank for first pass.
7. Address and Write Control sends Row Select to Memory Bank for first pass.
8. Write and Second Pass Control generates Write Function, Two Pass Function, and Write and No Two Pass.
9. Address and Write Control sends Column Address to Memory Bank for first pass.
10. Address and Write Control sends Column Select to Memory Bank for first pass.
11. Read Control generates Read Data Enable.
12. Address and Write Control sends Row Address to Memory Bank for second pass.
13. Memory Bank sends Read Data to Partial Write Network through Bank Read Data Mux and SECEDED Network.

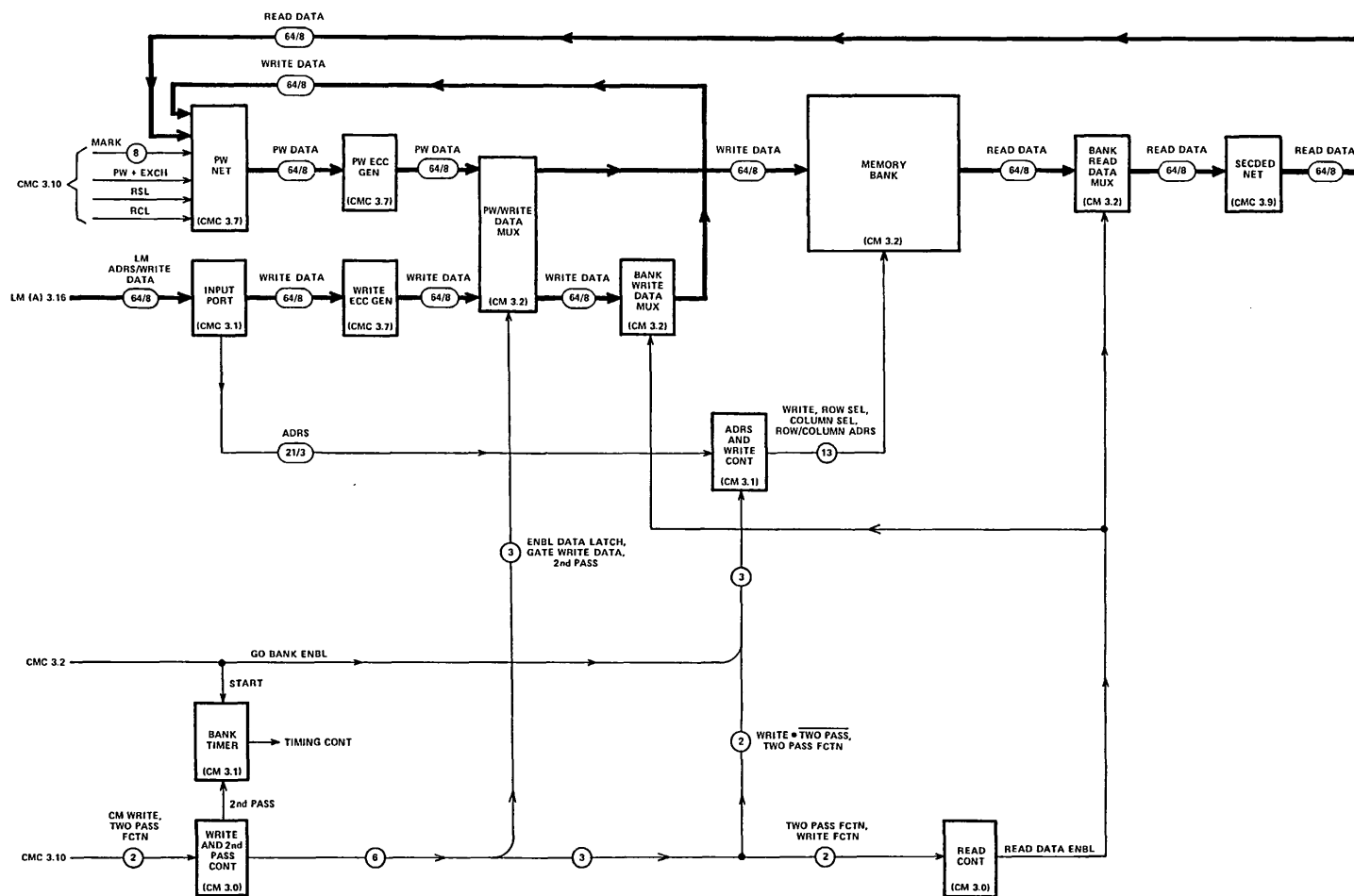


Figure 5-15. CM Partial Write

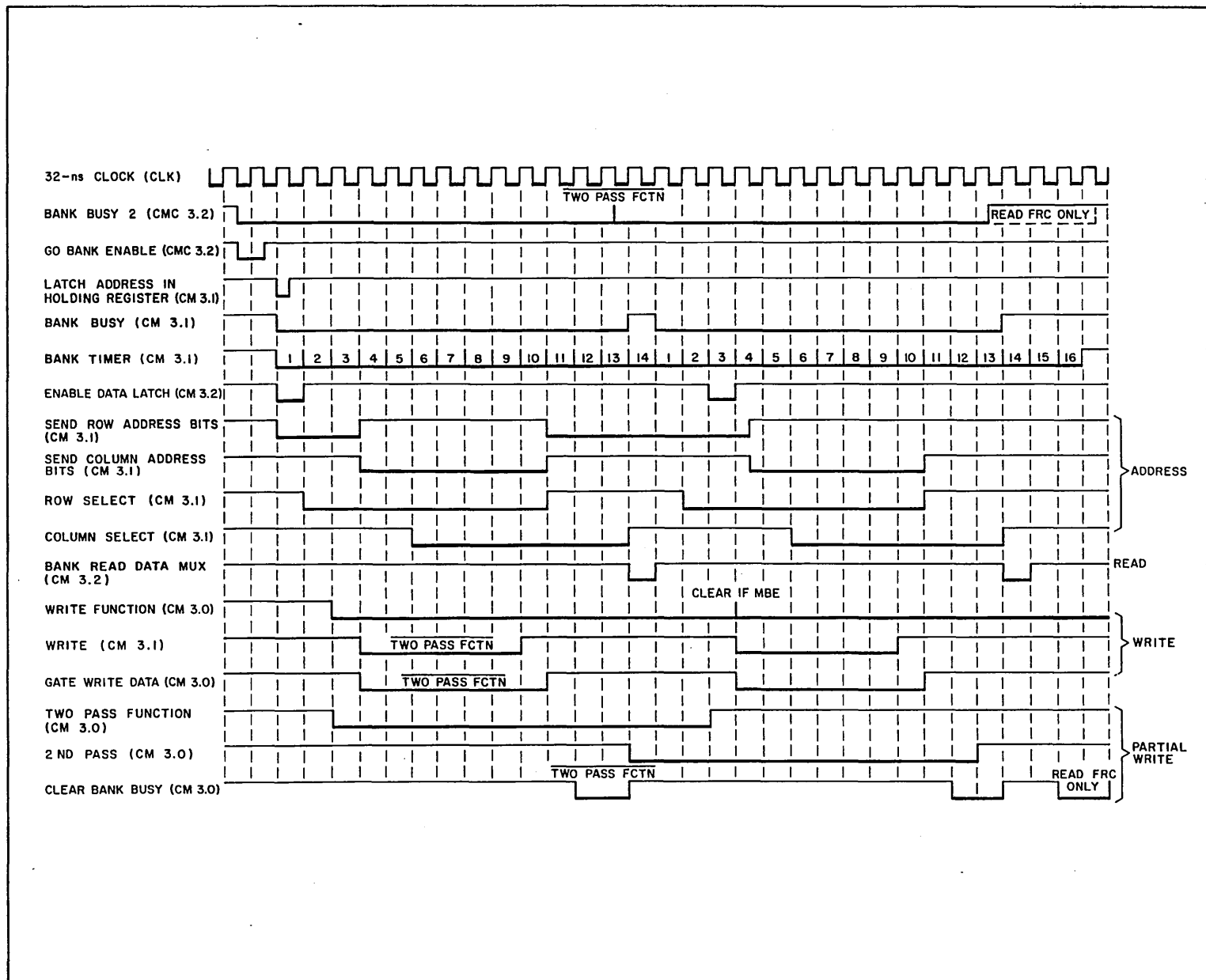


Figure 5-16. CM Bank Timing

14. Write and Second Pass Control generates Second Pass.
15. Partial Write/Write Data Mux sends Write Data to Partial Write Network through Bank Write Data Mux.
16. Partial Write Network forms Partial Write Data. Mark bits determine which Read Data byte(s) are to be modified. If a partial write (store byte) or exchange (swap) operation, marked Write Data byte(s) replace corresponding Read Data byte(s). If a read and set lock operation, logical OR of marked Read and Write Data byte(s) replace corresponding Read Data byte(s). If a read and clear lock operation, logical AND of marked Read and Write Data byte(s) replace corresponding Read Data byte(s).
17. Bank Timer starts counting for second pass.
18. Address and Write Control sends Row Select to Memory Bank for second pass.
19. Write and Second Pass Control generates Enable Data Latch.
20. Partial Write Network sends Partial Write Data to holding register in Partial Write/Write Data Mux through Partial Write ECC Generator.
21. Address and Write Control sends Write to Memory Bank.
22. Write and Second Pass Control generates Gate Write Data.
23. Partial Write/Write Data Mux sends Write Data to Memory Bank.
24. Address and Write Control sends Column Address to Memory Bank for second pass.
25. Address and Write Control sends Column Select to Memory Bank for second pass.
26. Clear Bank Busy Control (CM 3.0) sends Clear Bank Busy to Bank Busy Control (CMC 3.2). These circuits are not shown in figure 5-15.

## CENTRAL MEMORY (BS213-A, BS214-A, BS215-A)

### NOTE

The following description applies only to BS213-A, BS214-A, and BS215-A central memories (CMs). Refer to Central Memory (BS137-A) for a description of the BS137-A CM.

CM provides storage for LM (A), LM (B), IOU, and the Aux port. These ports access CM through CMC. The following information provides descriptions of:

- o Physical CM organization
- o CM addressing
- o Interface signals between CM and CMC
- o The major CM operations such as read, write, partial write, and long read cycle.

### PHYSICAL CM ORGANIZATION

#### Word Size

Each data word is made up of 72 bits (64 data plus eight error correction code or parity bits).

#### Chip size

Each chip contains 262,144 bits of MOS dynamic random access memory.

#### Memory Array Size

Each memory array is composed of 144 chips arranged in two rows of 72 chips. Therefore, each array stores a maximum of 522,288 x 72 bits (.5 M words).

#### Bank Size

Refer to CM 2.0. The CM has eight banks, each of which can have from one to four memory arrays, depending on the CM size.

#### Quadrant Size

A quadrant consists of one memory array (.5 M words) from each of the eight banks. It has a maximum capacity of 4 M words.



### CM Size

Refer to CM 2.0. The standard amount of CM is one half of quadrant 0 (2 M words). The first optional increment of CM is the other half of quadrant 0, expanding the CM to 4 M words. Additional increments add quadrants 1, 2, and 3, expanding the CM to 8 M, 12 M, and 16 M words.

### CM ADDRESSING

Refer to CM 1.0. Go Bank selects one of eight banks. Quad Sel selects one of the four quadrants (array boards) within a bank. Chip Sel selects one of two rows of a given memory array, thereby selecting the upper or lower half of the quadrant. The row and column addresses are then strobed into the memory chip by RAS (Row Address Strobe) and CAS (Column Address Strobe).

### INTERFACE SIGNALS FROM CMC

CMC sends address, data, and control signals to CM. Refer to CM 1.0 while reading the following descriptions of these signals.

### Go Bank

The presence of this signal initiates its respective bank for subsequent control signals.

### Address

This 27/4 bit path (27 address bits, 4 parity error bits) contains the address in CM being accessed. The address format is shown on CM 1.0.

### Write Data/Write Code, Write Parity

This 64/16 bit path (64 data bits plus 8 code and 8 parity bits) contains the data to be stored during a write operation. The 8 parity bits are not stored in CM but are returned to CMC during a partial write operation.

### Partial Write Data/Partial Write Code

This 64/8 bit path (64 data plus 8 code bits) contains the data to be stored during the second half of a partial write operation.

### Write

The presence of this signal indicates that the current request to the CM is for a write operation. The absence of this signal indicates that the current request to the CM is for a read operation.

### Multiple Bit Error (MBE)

The presence of this signal indicates that the SECEDED network in CMC detected a multiple-bit error in the data read from CM. If this signal occurs during a partial write operation, it prevents the writing of the modified data during the second cycle of the operation.

### Two Pass Function

The presence of this signal indicates that the current request to the CM is a partial write or an enable long cycle operation. Both of these operations require two bank cycles (passes).

### Enable Long Cycle

The presence of this signal delays the Bank 0 Clear Busy signal to CMC for 9 clock periods during the second cycle of a long read cycle operation.

### Refresh

This signal indicates that the current request to the CM is for a refresh cycle. This request takes priority over all other requests. To refresh memory, a read cycle is performed on each row address of each memory array.

## INTERFACE SIGNALS TO CMC

CM sends data and control signals to CMC. Refer to CM 1.0 while reading the following descriptions of these signals.

### Address PE Bytes 4-7

If an address parity error occurs during a write or partial write operation, one or more of the four address parity error bit outputs activates and prevents the write or partial write operation from being performed. If an Address parity error occurs during a read cycle, the cycle continues but the read data is discarded when it arrives at CMC.

### Write Data/Write Parity

This 64/8 bit path (64 data plus 8 parity bits) contains a copy of the write data sent to the Partial Write/Write Data Mux during a partial write operation. This data goes to the Partial Write Network in CMC.

### Read Data/Read Code

This 64/8 bit path (64 data plus 8 code bits) contains the data read from CM during a read operation.

#### Banks 0-7 Clear Busy

This signal indicates that a bank cycle is nearly complete. It clears the bank busy record in CMC for the bank that was accessed so that CMC can generate another Go Bank signal for this bank.

#### Address Bits 37-42 Inverted

These 6 bits indicate how much failing memory, if any, has been relocated. The status of these bits depends on the settings of SW-0 through SW-5 at location B14 in the CM. When a switch is set to up position, beginning with the lowest address memory blocks, it relocates a quantity of failing memory to the highest address memory blocks. The switches control the following amounts of failing memory:

<u>Switch</u>	<u>Failing Memory</u>
SW-0 (Bit 37)	64 MByte
SW-1 (Bit 38)	32 MByte
SW-2 (Bit 39)	16 MByte
SW-3 (Bit 40)	8 MByte
SW-4 (Bit 41)	4 MByte
SW-5 (Bit 42)	2 MByte

#### OPERATIONS

CM performs one of four operations depending on the conditions of the Go Bank, Write, and Two Pass Function signals from CMC. These four operations are:

- Read
- Write
- Partial Write
- Long Read Cycle

#### Read

While reading the description of this operation, refer to CM 1.0 and to the read timing diagram shown in figure 5-17.

#### Initialization:

Go Bank and Address arrive from CMC and are clocked in by the CM clock. Write and Two Pass Function are absent.

#### Response:

1. Bank Timer begins timing sequence to access selected address in CM. RAS and CAS latch the address into CM.

# NOTE

If an Adrs PE occurs during the read cycle, the cycle continues but the read data is discarded when it arrives at CMC.

2. Assuming that an Adrs PE does not occur during the read cycle, the read data is available at a CMC output port after 320 ns (10 clock periods).
3. Clear Busy arrives at CMC after 320 ns (10 clock periods), freeing the bank that was read for another operation.

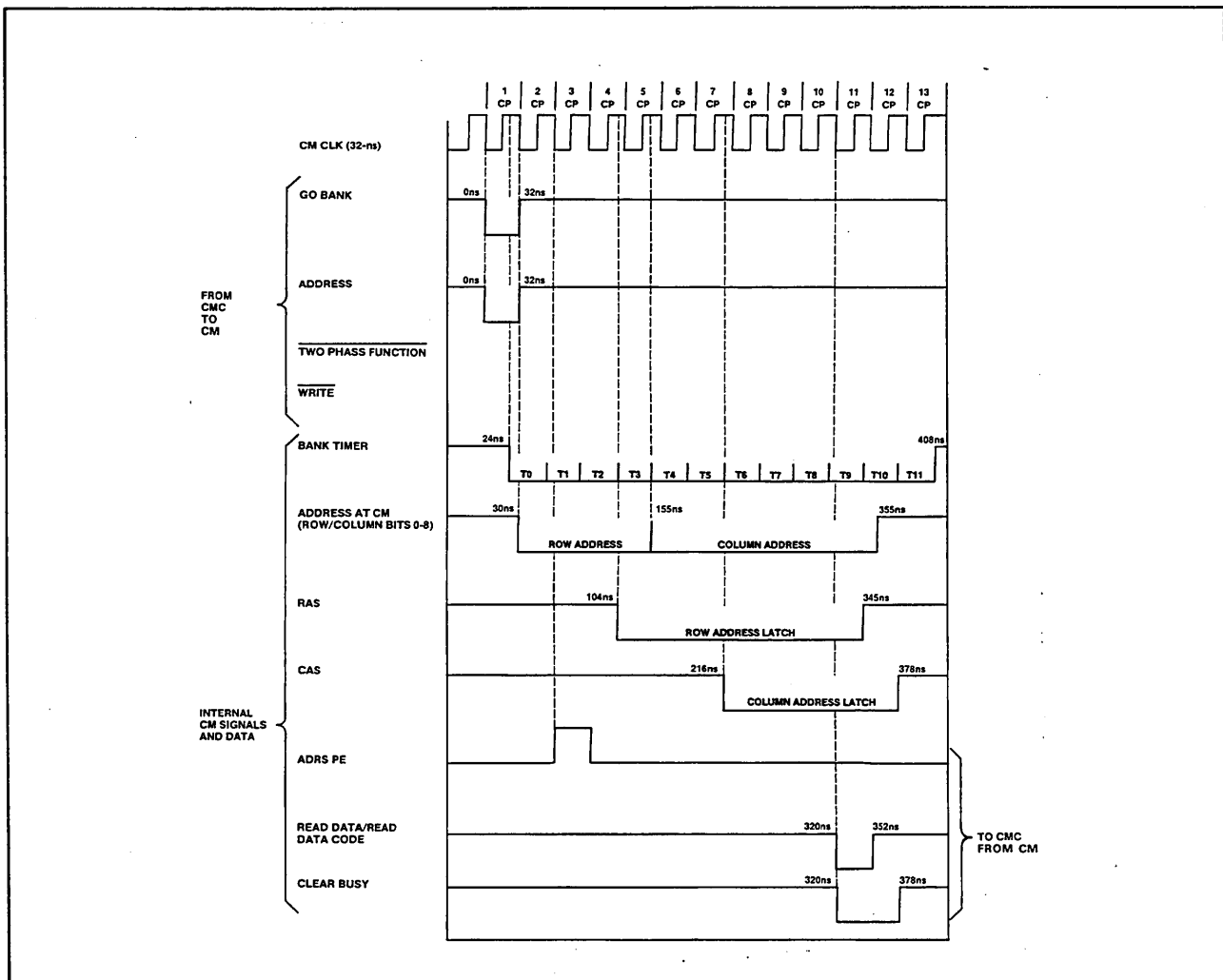


Figure 5-17. CM Read Timing

## Write

While reading the description of this operation, refer to CM 1.0 and to the write timing diagram shown in figure 5-18.

### Initialization:

Go Bank, Address, Write Data, and Write arrive from CMC and are clocked in by the CM clock. Two Pass Function is absent.

### Response:

1. Bank Timer begins timing sequence to access selected address in CM. RAS and CAS latch the address into CM.

### NOTE

If an Adrs PE occurs during the write cycle, the write cycle converts to a read cycle and returns the read data to CMC.

2. Assuming that an Adrs PE does not occur, Write Data/Write Data Code is stored in CM after 170 ns (approximately 5.25 clock periods).
3. Clear Busy arrives at CMC after 320 ns (10 clock periods), freeing the bank that was written into for another operation.

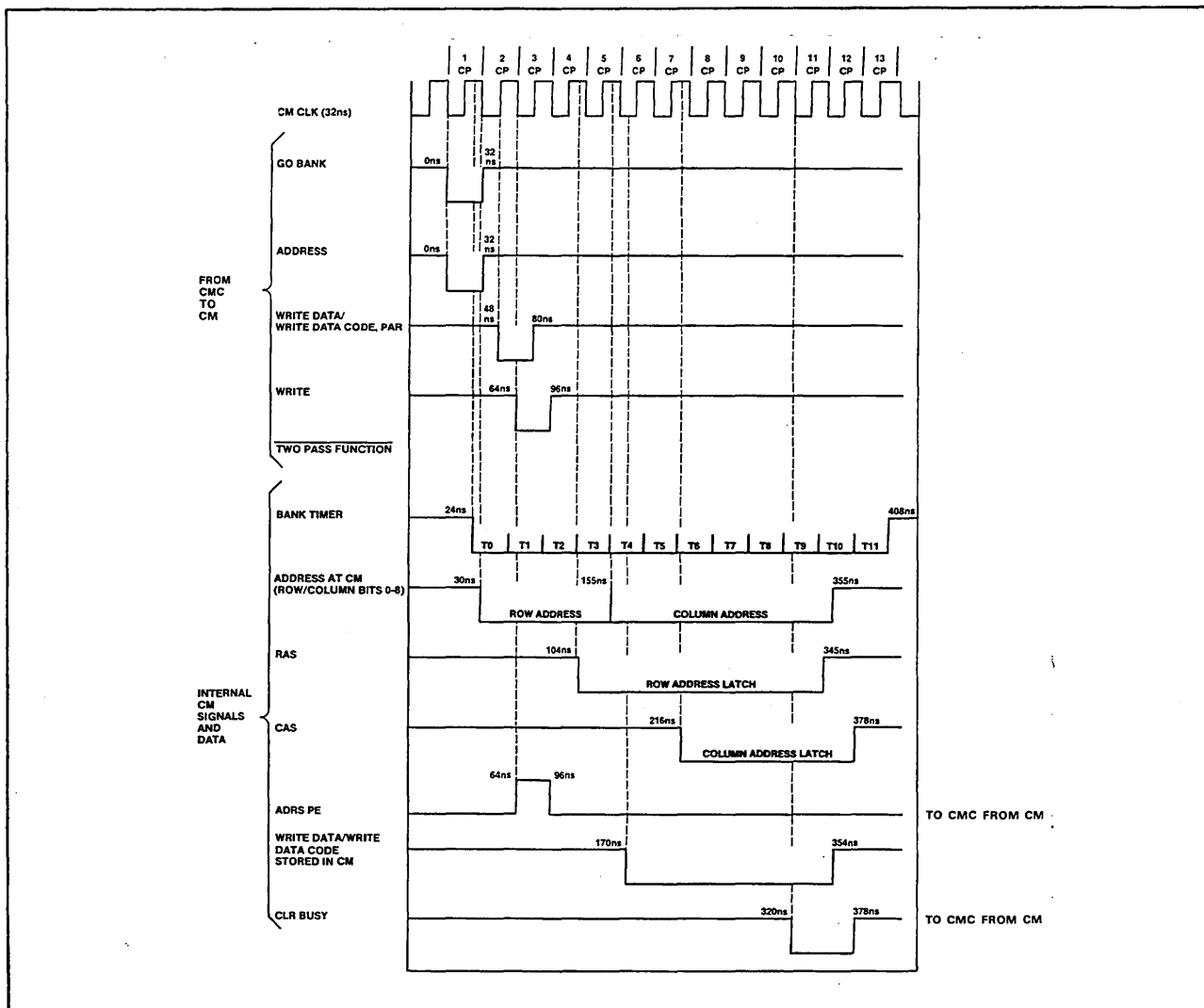


Figure 5-18. CM Write Timing

### Partial Write

The purpose of a partial write operation is to coordinate write requests so that data at a location is not written over before it is read and evaluated. This requires two timing cycles (passes). During the first pass, the selected location in CM is read. Write data is also sent to CM but is blocked and returned to CMC with write parity bits at the end of the first pass. If an error does not occur during the first pass, the data read from CM is evaluated. Based on the status of the read data, the write data is modified, if necessary, to become partial write data. The partial write data is written into the original CM location during the second pass. While reading the detailed description of this operation, refer to CM 1.0 and to the partial write timing diagram shown in figure 5-19.

#### First Pass Initialization:

1. Go Bank, Address, Write Data, Write, and Two Pass Function arrive from CMC and are clocked in by the CM clock.
2. Write data is blocked from being written into CM by the presence of Two Pass Function.

#### First Pass Response:

1. Bank Timer begins first pass timing sequence to access selected address in CM. RAS and CAS latch the address into CM.
2. Selected address is read and read data is returned to CMC after 320 ns (10 clock periods).
3. Write data that was blocked from being written into CM is returned with parity bits to CMC after 352 ns (11 clock periods).

#### Second Pass Initialization:

1. A second Go Bank (not shown on timing diagram) is generated in CM after 384 ns (12 clock periods) to restart the CM Bank Timer for the second pass.
2. Partial write data and code arrive from CMC and are clocked in by the CM clock.

#### Second Pass Response:

1. Bank Timer begins second pass timing sequence to access original address in CM. RAS and CAS latch the address into CM.

#### NOTE

- If a multiple bit error (MBE) is detected in the data read during the first pass, MBE is clocked in from CMC at the beginning of the second pass. MBE blocks the writing of partial write data into the selected CM address by converting the second pass to another read cycle. However, the read data is also blocked since another bank may be reading valid data into the read data output bus. A conflict would corrupt the valid data.
  - If an Adrs PE is detected during the first pass, the second pass becomes another read cycle. However, the read data is blocked since another bank may be reading valid data into the read data output bus. A conflict would corrupt the valid data.
2. Assuming that a MBE or Adrs PE did not occur during the first pass, partial write data and code are stored in CM after 554 ns (approximately 17.5 clock periods).
  3. Clear Busy arrives at CMC after 699 ns (approximately 22 clock periods), freeing the bank that the partial write data was written into for another operation.



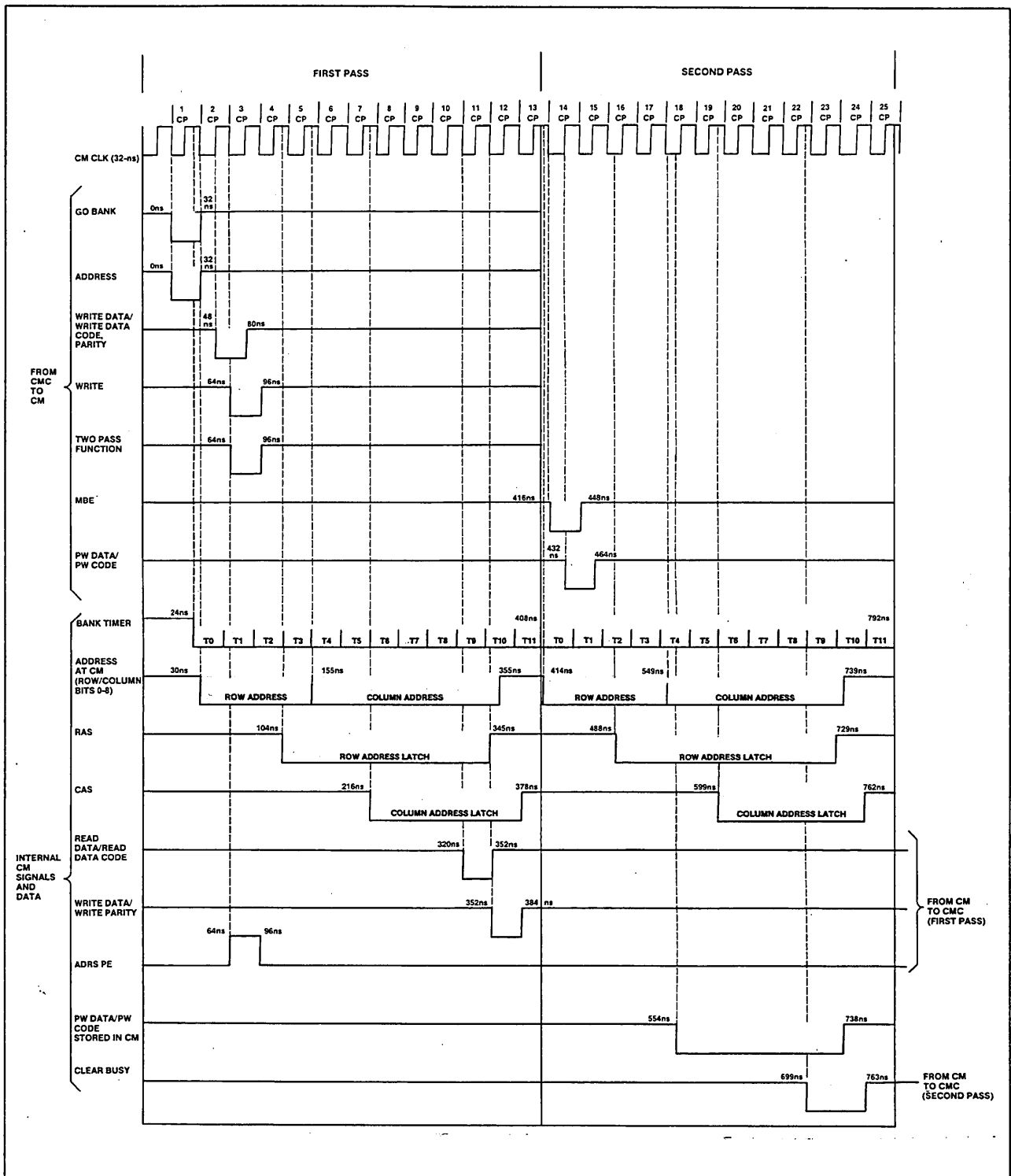


Figure 5-19. CM Partial Write Timing

### Long Read Cycle

This operation tests the Free Running Counter (FRC) in CMC for proper operation. This requires two memory cycles (passes). A memory location in CM is read twice, once during the first pass and again during the second pass. The read data from each pass is returned to CMC but is not used. While reading the description of this operation, refer to the Long Read Cycle timing diagram shown in figure 5-20.

#### First Pass Initialization:

Go bank, Address, and Two Pass Function arrive from CMC and are clocked in by the CM clock. Write is absent.

#### First Pass Response:

1. Bank Timer begins timing sequence to access selected address in CM.

#### NOTE

If an Adrs PE occurs during the first pass, the operation continues but the data is discarded when it arrives at CMC.

2. The selected address is read and read data is returned to CMC after 320 ns (10 clock periods).

#### Second Pass Initialization:

1. A second Go Bank (not shown on timing diagram) is generated in CM after 384 ns (12 clock periods) to restart the CM Bank Timer for the second pass.
2. Enable Long Cycle arrives from CMC and is clocked in by the CM clock.

#### Second Pass Response:

1. Bank Timer begins second pass timing sequence to access original address in CM. RAS and CAS latch the address into CM.

#### NOTE

If an address PE occurred during the first pass, the second pass read data is discarded when it arrives at CMC.

2. The selected address is read and read data is returned to CMC after 704 ns (22 clock periods).
3. Clear Busy arrives at CMC after 992 ns (31 clock periods), freeing the bank that was read for another operation.

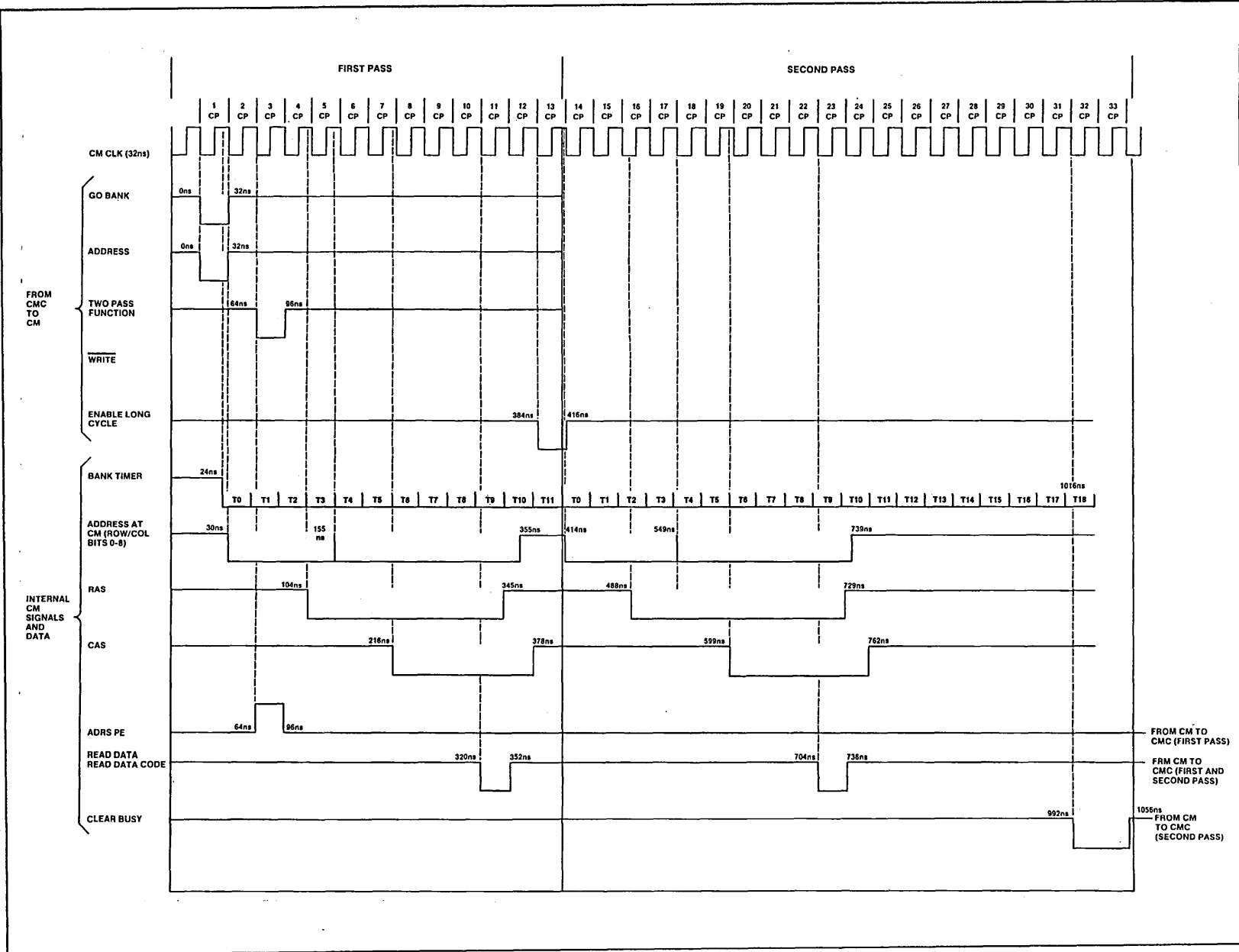


Figure 5-20. CM Long Read Cycle Timing



## SECTION 6

### INSTRUCTION COMPLETION CONTROL (ICC)



---

ICC accumulates and interprets a variety of conditions which may arise between the time an instruction arrives in IF and the time the instruction completes its execution cycle. These conditions include hardware errors, software exceptions, and events occurring in the external environment, such as power failures.

ICC typically determines at the point of no return (PONR) the appropriate measure to be taken in response to a detected error or exception. PONR occurs when an active PONR bit from the General Micrand enters ICP rank 50. PONR generally occurs when the first operating register in the Register File or memory location is about to be written during an instruction's execution.

Time-critical errors and exceptions (ones which would invalidate a process if it is allowed to continue) that occur before PONR terminate instruction execution at PONR. Conditions which occur before PONR and are noncritical or occur after PONR are evaluated at the next PONR (typically during execution of the next instruction). Terminating routines include instruction retry, exchange or trap interrupts, and CP halt.

#### ERRORS, EXCEPTIONS

Detected errors and exceptions enter three registers in ICC for subsequent evaluation. The registers are the User Condition Register (UCR), Monitor Condition Register (MCR) and C170 Exit Mode FFs (ECR). The UCR and MCR accommodate four classes of information. They are monitor conditions (typically system evaluation of user code), system conditions (developments in the system which are independent of user processes), user conditions (usually arithmetic exceptions) and system status indicators. The ICC establishes an order for handling these conditions, with the contents of the MCR usually having priority over the contents of the UCR. The MCR contains all system conditions, status indicators and most of the monitor conditions. The UCR contains all user conditions and some monitor conditions. The UCR receives the monitor conditions selected to initiate trap interrupts in job mode if traps are enabled. These monitor conditions and certain user conditions (Process Interval Timer, Free Flag, and Keypoint) may be activated only by microcode-directed (live register) writes.

Figures 6-1 and 6-2 show the contents of the MCR and UCR, respectively.

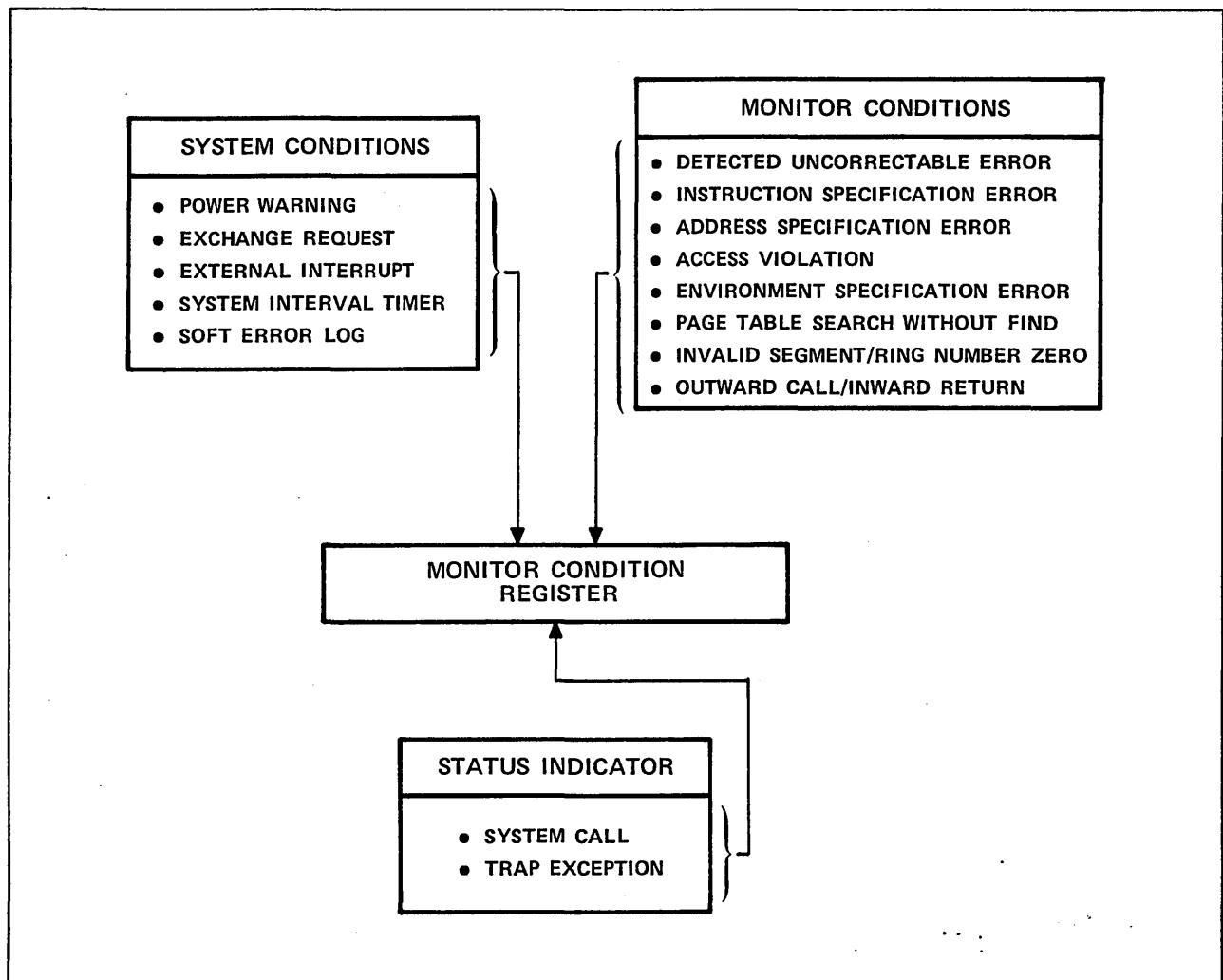


Figure 6-1. MCR Conditions



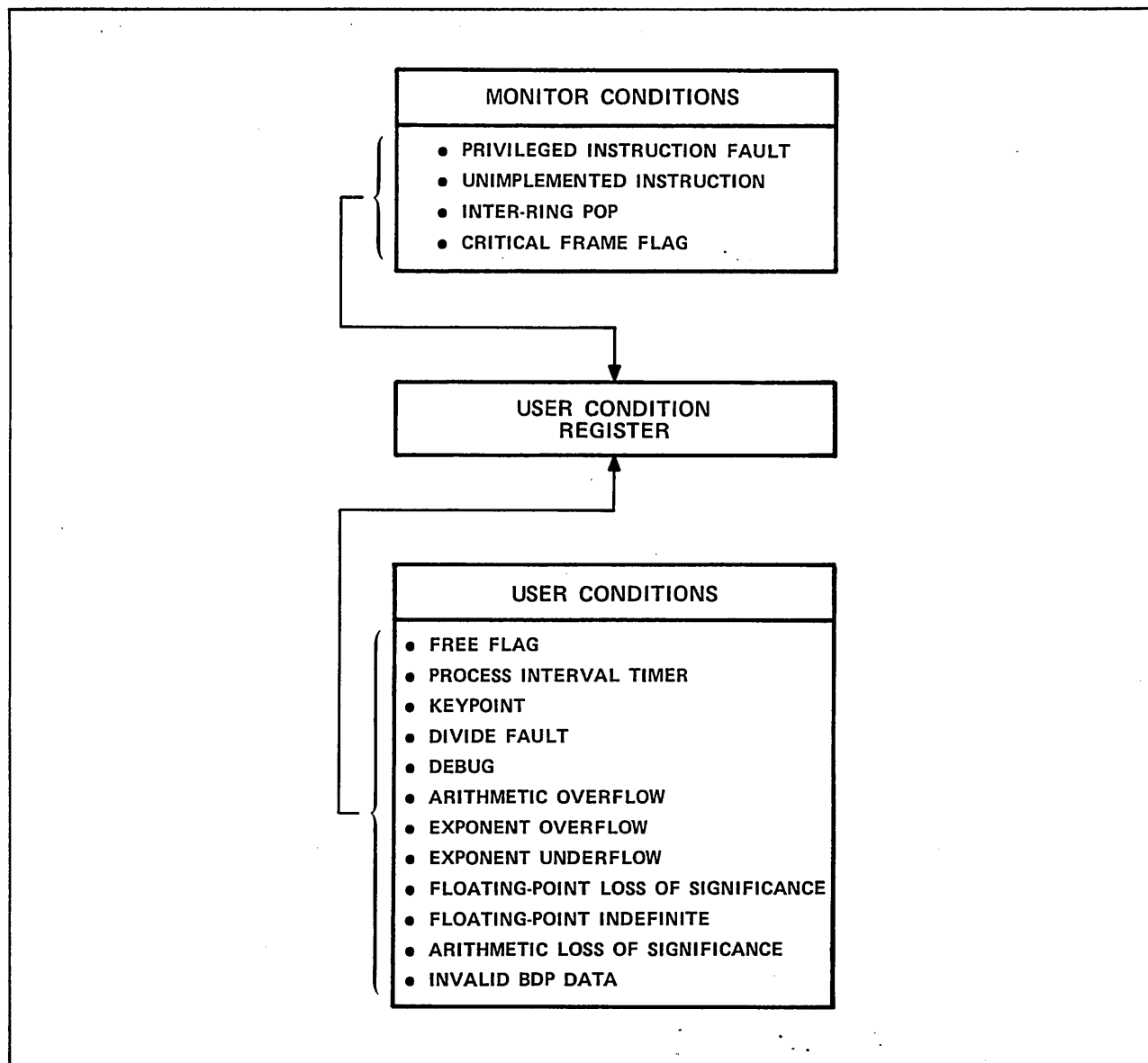


Figure 6-2. UCR Conditions

The C170 Exit Mode Condition FFs receive a combination of user and monitor conditions associated with C170 operations. Included are C170 Illegal Instruction, C170 Infinite Status, C170 Indefinite Status, Rank 41 Address Out of Range (AOR), and Rank 41 C170 Operand AOR.

All three registers described previously consist of two stages, one associated with execution Time 50 of the micrand containing Rank 50 PONR or Rank 50 Clock Condition Register (CCR) and the other with execution Time 60. The Before PONR UCR, Before PONR MCR, and Rank 50 C170 Exit Mode Condition FFs correspond to Time 50. The MCR, UCR, and C170 Exit Mode Condition FFs correspond to Time 60.

#### CCR, PONR

Rank 50 CCR and Rank 50 PONR control movement of exceptions and errors through the rank 50 and rank 60 condition registers. Both bits are contained in the General Micrand and arrive in ICC from ICP rank 50. Rank 50 PONR or Rank 50 CCR enables the contents of rank 50 condition registers to enter the rank 60 condition registers at the same time it clears the rank 50 registers. Rank 50 PONR also enables ICC to initiate an exchange, trap, or halt interrupt or instruction retry if the condition sensed is time critical and the instruction is in the Before PONR state.

Refer to figure 6-3 for examples of the typical use of CCR and PONR within single and multiple micrand instructions.

Instruction 1 represents the case in which CCR from the preceding instruction establishes a Before PONR condition in ICC. Errors or exceptions detected while Micrands 1 through 3 are executing enter the Before PONR condition registers (rank 50). Errors or exceptions that are time critical enter Condition to Mask Comparators and Interrupt Detection Control directly from the Before PONR MCR and Before PONR UCR. Included are Before PONR MCR PDM, Before PONR MCR Bits 3, 4, 6 through 9, 12, 13, and Before PONR UCR Bits 7, 9, 13 through 15. Rank 50 Exit Mode Condition Bits 3 and 4 likewise have this critical character and enter C170 Exchange Interrupt Control for interrupt sensing.

When Micrand 3 enters ICP rank 50, Rank 50 PONR gates Before PONR conditions to the MCR, UCR, and C170 Exit Mode condition FFs. Micrand 3 also enables an interrupt or halt routine if a time-critical condition exists in rank 50. Rank 50 PONR clears the rank 50 condition registers at the same time the Before PONR condition bits enter rank 60. The rank 60 bits (MCR and UCR Bits 0 through 15 and Exit Mode Condition Bits 0 through 4) may be read by OPI, providing a record of any interrupt condition to the exchange package in the Register File.

A noncritical Before PONR condition entering the rank 60 condition register remains there until a subsequent Rank 50 PONR enables interrupt sensing.

Any interrupting condition in rank 60, time critical or not, prevents most After PONR conditions associated with Micrands 4 through 6 from entering the rank 50 condition registers until that bit is cleared in rank 60. Some of these rank 60 conditions latch rank 50 automatically. Others must have their mask bits set. Interrupt software must clear the bit.

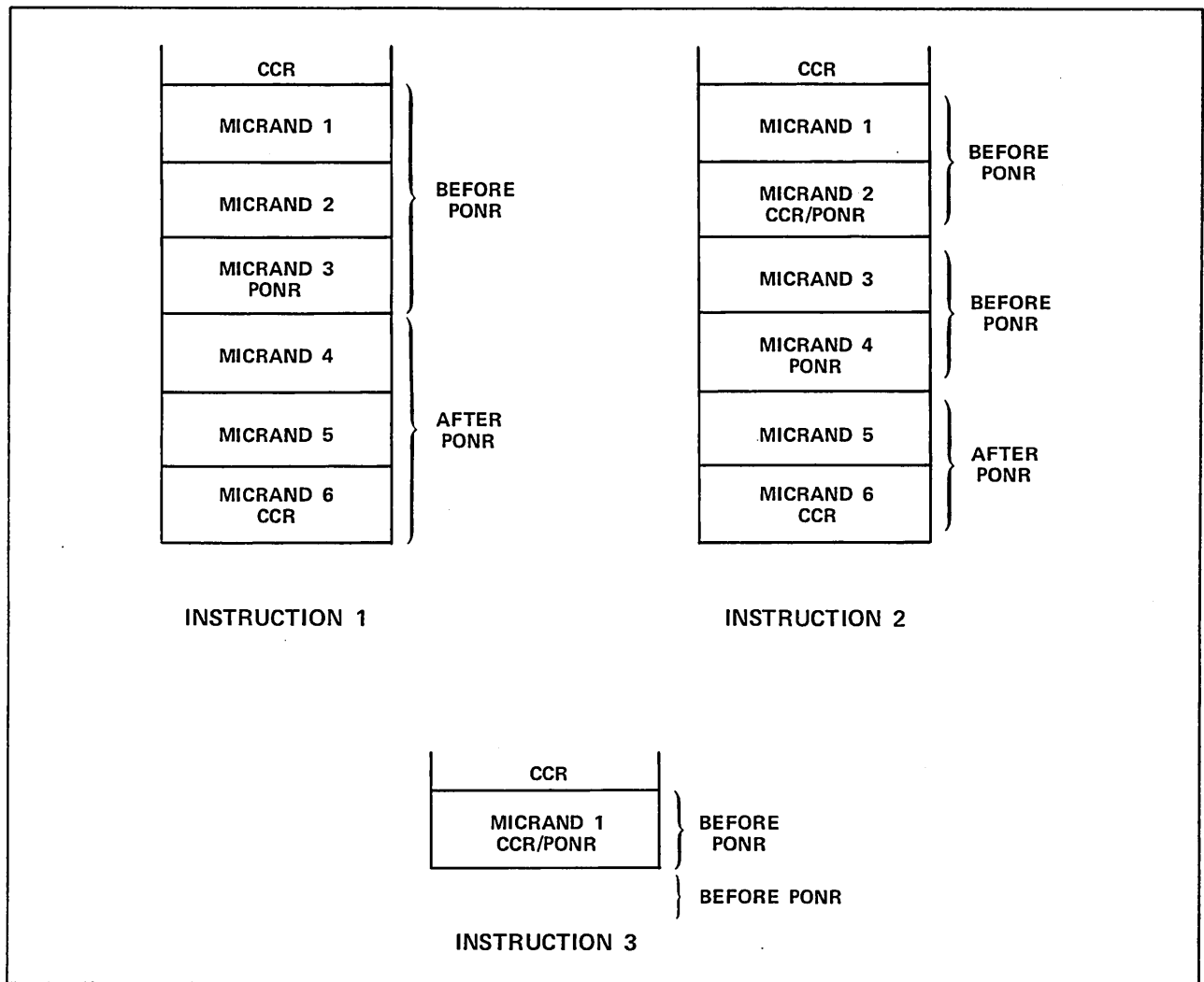


Figure 6-3. CCR, PONR Implementation

If the Before PONR condition bits are all inactive at the time they enter the rank 60 condition registers (or in some cases, the associated mask bits are clear), After PONR conditions enter the rank 50 condition registers. When Micrand 6 enters ICP rank 50, Rank 50 CCR enables After PONR conditions to the MCR, UCR, and C170 Exit Mode Condition FFs. Rank 50 CCR clears the rank 50 condition registers at the same time the After PONR conditions enter rank 60. If one of these bits is active (and, if required, the associated mask bit is set), the bit prevents most of the next instruction's Before PONR conditions from entering the rank 50 condition registers before that instruction's PONR arrives to enable an interrupt. The rank 60 condition clears at the conclusion of the interrupt. If all the After PONR conditions are inactive, the next instruction's Before PONR conditions enter the rank 50 condition registers.

Instruction 2 is like instruction 1 except that Micrand 2 contains CCR/PONR, a test interrupt feature. The combination of Rank 50 CCR and Rank 50 PONR maintains the instruction in the Before PONR state at the same time it enables sensing of time-critical conditions arriving during Micrands 1 and 2. CCR/PONR latches all condition bits associated with Micrands 1 and 2 into rank 60 registers at the same time CCR/PONR clears them from rank 50 registers. CCR/PONR does not signify the arrival of point of no return. CCR/PONR is inserted only for the purpose of triggering interrupts prior to PONR.

Instruction 3 shows the use of CCR and PONR in a single-micrand instruction. CCR/PONR is essentially a test interrupt routine. CCR/PONR enables sensing of time-critical Before PONR conditions connected with Micrand 1 and preserves the Before PONR state to accommodate the instruction that follows.

#### CONDITION DESCRIPTIONS

The errors and exceptions evaluated by ICC are defined in tables 6-1 and 6-2.

Table 6-1. MCR Conditions and Descriptions (Sheet 1 of 5)

Bit	Name	Description
0	Detected Uncorrectable Error	Indicates an uncorrectable error condition has been detected during a CP or memory operation. Among the errors activating this critical failure bit are parity errors in data sent from the CP to memory, memory SECEDED errors, parity errors in data sent from memory to the CP, and parity errors in selected CP data. These errors, termed processor detected malfunctions (PDMs), may initiate instruction retry if they occur before the point of no return. PDMs occurring after the point of no return indicate a damaged process.
1	Unimplemented	If the PDM indicates segment map, page map, or cache memory contains invalid data, an instruction retry, trap, or exchange interrupt purges the data.

Table 6-1. MCR Conditions and Descriptions (Sheet 2 of 5)

Bit	Name	Description																												
2	Power Warning	Indicates a system power failure, a local 50-Hz 160-Hz power failure, high-temperature fault, or condensing unit fault. Bit also indicates low-temperature fault or blower fault if these conditions are not considered long warning failures.																												
3	Instruction Specification Error	<p>Indicates the following:</p> <ul style="list-style-type: none"><li>During C180 Isolate/Insert (AC, AD, and AE) instructions that the sum of leftmost position designator plus length designator is greater than 63.</li><li>During C180 BDP instruction that the length specified by data descriptor L field exceeds maximum length for applicable data type.</li><li>That data type fields in source and/or destination data descriptors are invalid during the following C180 BDP instructions:</li></ul> <table><thead><tr><th>Op Code</th><th>Instruction</th></tr></thead><tbody><tr><td>70</td><td>Decimal Sum</td></tr><tr><td>71</td><td>Decimal Difference</td></tr><tr><td>72</td><td>Decimal Product</td></tr><tr><td>73</td><td>Decimal Quotient</td></tr><tr><td>74</td><td>Decimal Compare</td></tr><tr><td>75</td><td>Numeric Move</td></tr><tr><td>E4</td><td>Decimal Scale</td></tr><tr><td>E5</td><td>Decimal Scale Rounded</td></tr><tr><td>ED</td><td>Edit</td></tr><tr><td>F4</td><td>Calculate Subscript and Add</td></tr><tr><td>F9</td><td>Move Immediate Data</td></tr><tr><td>FA</td><td>Compare Immediate Data .</td></tr><tr><td>FB</td><td>Add Immediate Data</td></tr></tbody></table> <ul style="list-style-type: none"><li>Execution of C180 Calculate Subscript and Add (F4) instruction when bits 61-63 of the PVA used to access subscript range table (SRT) do not equal 0.</li><li>Execution of a C180 Program Error (00) instruction.</li><li>Execution of a C180 Copy to State Register (0F) instruction or a Branch on Condition Register (9F) instruction when execute access is restricted to the C180 monitor mode and the CP is not in that mode.</li><li>Execution of a C180 Call (B0/B5) instruction when the number of the last A Register field (A Terminate) is less than 2.</li></ul>	Op Code	Instruction	70	Decimal Sum	71	Decimal Difference	72	Decimal Product	73	Decimal Quotient	74	Decimal Compare	75	Numeric Move	E4	Decimal Scale	E5	Decimal Scale Rounded	ED	Edit	F4	Calculate Subscript and Add	F9	Move Immediate Data	FA	Compare Immediate Data .	FB	Add Immediate Data
Op Code	Instruction																													
70	Decimal Sum																													
71	Decimal Difference																													
72	Decimal Product																													
73	Decimal Quotient																													
74	Decimal Compare																													
75	Numeric Move																													
E4	Decimal Scale																													
E5	Decimal Scale Rounded																													
ED	Edit																													
F4	Calculate Subscript and Add																													
F9	Move Immediate Data																													
FA	Compare Immediate Data .																													
FB	Add Immediate Data																													

Table 6-1. MCR Conditions and Descriptions (Sheet 3 of 5)

Bit	Name	Description																		
4	Address Specification Error	<p>Indicates an attempt by the CP to use an improper address.</p> <ul style="list-style-type: none"><li>Data word address with nonzero bits 61-63 generated by the following instructions:</li></ul> <table><thead><tr><th><u>C180 Opcode</u></th><th><u>Instruction</u></th></tr></thead><tbody><tr><td>04</td><td>Return</td></tr><tr><td>B5</td><td>Call Indirect</td></tr><tr><td>80</td><td>Load Multiple</td></tr><tr><td>81</td><td>Store Multiple</td></tr><tr><td>82</td><td>Load Word</td></tr><tr><td>83</td><td>Store Word</td></tr><tr><td>F4</td><td>Calculate Subscript and Add</td></tr><tr><td>-</td><td>Debug List Pointer</td></tr></tbody></table> <ul style="list-style-type: none"><li>Instruction parcel address with nonzero bits 62 and 63 generated by C180 Intersegment Branch (2F) instruction.</li><li>Any PVA with nonzero bit 32.</li><li>Any instruction PVA (parcel address) with a nonzero bit 63.</li></ul> <p>An address specification error may also be detected during,</p> <ul style="list-style-type: none"><li>C180 decimal arithmetic (70 to 75, E4, and E5) instructions.</li><li>C180 Move Immediate Data (F9) instruction.</li><li>C180 Convert from Floating-Point to Integer (3B) instruction.</li><li>C180 Test and Set Page (16) instruction.</li></ul>	<u>C180 Opcode</u>	<u>Instruction</u>	04	Return	B5	Call Indirect	80	Load Multiple	81	Store Multiple	82	Load Word	83	Store Word	F4	Calculate Subscript and Add	-	Debug List Pointer
<u>C180 Opcode</u>	<u>Instruction</u>																			
04	Return																			
B5	Call Indirect																			
80	Load Multiple																			
81	Store Multiple																			
82	Load Word																			
83	Store Word																			
F4	Calculate Subscript and Add																			
-	Debug List Pointer																			
5	Exchange Request	<p>Indicates the CP has received a C170 exchange request from the IOU, which has executed one of the following Instructions: Exchange Jump (002600), Monitor Exchange Jump (002610), or Monitor Exchange Jump to MA (002620). When the CP is in the C180 state, the operating system must switch the CP to C170 job mode before the exchange can take place.</p>																		
6	Access Violation	<p>Indicates a requested memory access has been blocked because it does not have the required permission. Specific violations are memory read, write, or execute access which are not permitted or which are not within ring limits. Other violations are a call via a Code Based Pointer which is not in Binding Section segment, key/lock violations, or call from process outside call ring limit.</p>																		

Table 6-1. MCR Conditions and Descriptions (Sheet 4 of 5)

Bit	Name	Description															
7	Environment Specification Error	<p>If detected during a call, return, or pop instruction, or during an exchange or trap operation, indicates:</p> <ul style="list-style-type: none"> <li>• A mismatch between Virtual Machine Capability List (VMCL) and the Virtual Machine ID (VMID) obtained from the Code Base Pointer on a C180 Call Indirect (B5) instruction.</li> <li>• A mismatch between VMCL and the VMID obtained from the Stack Frame Save Area on a C180 Return (04) instruction.</li> <li>• Initial A2 (previous save area pointer) not equal to A0 (dynamic space pointer) in the Stack Frame Save Area on a C180 Return (04) or Pop (06) instruction.</li> <li>• In the previous stack frame descriptor, the value of the field designating the last A Register to be loaded is less than 2 (A Terminate less than 2) on a C180 Return (04) instruction.</li> <li>• A mismatch between VMCL and VMID obtained from the exchange package during an exchange operation.</li> <li>• A mismatch between VMCL and the VMID obtained from the Code Base Pointer during a trap interrupt.</li> <li>• External procedure flag not set in the Code Base Pointer during a trap interrupt.</li> </ul>															
8	External Interrupt	<p>Indicates execution of a C180 Processor Interrupt (03) instruction in the interrupted CP, or in another CP in a multiprocessor system through the CM port specified by the instruction. Bit also may indicate the IOU is executing an Interrupt Processor (1026X) instruction. Port designation codes associated with either of these instructions appear below. Port 3 is connected to the IOU; Port 2 is an auxiliary port.</p> <table> <tr> <th>CMC Port</th><th>P3 Instruction <math>X_k</math> Bit</th><th>IOU 1026X Instruction d Field Decode</th></tr> <tr> <td>0</td><td>63</td><td>01</td></tr> <tr> <td>1</td><td>62</td><td>02</td></tr> <tr> <td>2</td><td>61</td><td>04</td></tr> <tr> <td>3</td><td>60</td><td>08</td></tr> </table>	CMC Port	P3 Instruction $X_k$ Bit	IOU 1026X Instruction d Field Decode	0	63	01	1	62	02	2	61	04	3	60	08
CMC Port	P3 Instruction $X_k$ Bit	IOU 1026X Instruction d Field Decode															
0	63	01															
1	62	02															
2	61	04															
3	60	08															

Table 6-1. MCR Conditions and Descriptions (Sheet 5 of 5)

Bit	Name	Description
9	Page Table Search Without Find	Indicates the Page Table entry was not found in the Page Table at entry address or at any of the following 31 addresses. Conversion of the SVA into an RMA is not possible.
10	System Call	Indicates that an executed C180 Exchange (02) instruction from job to monitor mode has caused an exchange interrupt.
11	System Interval Timer	Indicates System Interval Timer has decremented to zero.
12	Invalid Segment	Indicates: <ul style="list-style-type: none"> <li>• A PVA could not be translated into an RMA because the Segment Table Length was exceeded or the Segment Descriptor was invalid.</li> <li>• A C180 Call (B0 or B5) instruction attempted to execute with a Code Base Pointer Ring Number equal to zero.</li> <li>• An A Register was loaded with a PVA with Ring Number equal to zero during a load A (80, 84, A0), Return (04), or Pop (06) instruction.</li> </ul>
13	Outward Call/Inward Return	Indicates an outward call or inward return has been attempted by the CP. An outward call is to a procedure with a higher Ring Number than the current P Ring Number. In an inward return is to a procedure with a lower Ring Number than the current P Ring Number.
14	Soft Error Log	Indicates that an error has been detected and corrected by the hardware, including the following: <ul style="list-style-type: none"> <li>• A corrected error in CM for the port used by this CP.</li> <li>• A corrected hardware malfunction in the CP.</li> <li>• A parity error for an instruction following an instruction that has caused an unbranch or interrupt. The parity error would cause an interrupt if the instruction were permitted to execute.</li> </ul>
15	Trap Exception	Indicates a fault during a trap interrupt operation. In such a case, at least one other MCR bit indicates the cause of the trap exception.



Table 6-2. UCR Conditions and Descriptions (Sheet 1 of 2)

Bit	Name	Description
0	Privileged Instruction Fault	Indicates attempted execution of a locally privileged instruction in other than a locally privileged or globally privileged execution mode. Also indicates attempted execution of a globally privileged instruction in other than a globally privileged execution mode. The C170 Trap (017) instruction sets the bit to initiate a trap routine.
1	Unimplemented Instruction	Indicates that an instruction not implemented in the CP attempts to execute. C170 compare/move (464-467) instructions also initiate this interrupt.
2	Free Flag	Is normally set by software in an exchange package in CM and causes an immediate trap interrupt after an exchange operation loads it into the CP. Software uses this flag. Hardware does not set the bit.
3	Process Interval Timer	Indicates the Process Interval Timer has decremented to zero.
4	Inter-Ring Pop	Indicates when an attempt is made to pop a stack frame in one ring by means of a C180 Pop (06) instruction executing in a different ring.
5	Critical Frame Flag	Indicates the CP's attempt to execute a pop or return instruction from a critical stack frame.
6	Keypoint	Indicates the CP is executing an enabled C180 Keypoint (B1) instruction to permit the software to collect performance data at this point in the program.
7	Divide Fault	Indicates the CP has detected a divisor equal to zero during execution of a C180 integer quotient (23, 27, and 73) instruction. For the C180 floating-point quotient (33 and 37) instructions, the CP detects a divide fault if the coefficient of the divisor is a nonstandard value of zero, or is unnormalized and can be divided into the coefficient of the dividend by a factor greater than or equal to 2.
8	Debug	Indicates a debug match.
9	Arithmetic Overflow	Indicates one of the following conditions: <ul style="list-style-type: none"> <li>• C180 integer sum (10, 20, 24, 28, 8A, and 8B) instructions when augend and addend have same signs but sum has opposite sign.</li> <li>• C180 integer difference (11, 21, 25, and 29) instructions when minuend and subtrahend have opposite signs but difference has sign opposite of minuend sign.</li> </ul>

Table 6-2. UCR Conditions and Descriptions (Sheet 2 of 2)

Bit	Name	Description
		<ul style="list-style-type: none"> <li>• C180 half-word integer product (22 and 8C) instructions when most significant 32 bits of intermediate product are not equal to sign bit.</li> <li>• C180 integer product (26 and B2) instructions when leftmost 64 bits of intermediate product are not equal to sign bit.</li> <li>• C180 Half-word Integer Quotient (23) instruction when <math>-2^{31}</math> is divided by <math>-2^0</math>.</li> <li>• C180 Integer Quotient (27) instruction when <math>-2^{64}</math> is divided by <math>-2^0</math>.</li> <li>• C180 decimal arithmetic (70 to 73) instructions when result length exceeds destination field length.</li> <li>• C180 Add Immediate Data (FB) instruction when source or destination field data descriptors specify invalid data type.</li> </ul>
10	Exponent Overflow	Indicates that a C180 floating-point comparison or arithmetic instruction has produced an exponent with an actual value between $2^{4096}$ and $2^{12,187}$ .
11	Exponent Underflow	Indicates that a C180 floating-point arithmetic instruction has produced an intermediate exponent value between $-2^{4096}$ and $-2^{12,187}$ .
12	Floating-Point Loss of Significance	Indicates that a floating-point arithmetic instruction has produced an intermediate result with an overflow bit and coefficient of all zeros.
13	Floating-Point Indefinite	Indicates that a floating-point arithmetic instruction has produced a final nonstandard indefinite result.
14	Arithmetic Loss of Significance	Indicates that significant digit(s) in the result are truncated or not stored in CM during execution of the following instructions: <ul style="list-style-type: none"> <li>• C180 decimal arithmetic (70 to 75, E4 and E5) instructions.</li> <li>• C180 Move Immediate Data (F9) instruction.</li> <li>• C180 Convert from Floating-Point to Integer (3B) instruction.</li> </ul>
15	Invalid BDP Data	Indicates the CP had detected an invalid decimal digit during execution of BDP decimal numeric, calculate subscript and add, compare immediate data, move immediate data, edit, or convert floating-point to integer instruction.

### Interrupt Condition Groups

The grouping of condition bits in tables 6-3 and 6-4 is based on how the conditions are generated and when instructions are interrupted.

Table 6-3. Interrupt Condition Groups (Sheet 1 of 2)

Group	Interrupt Condition
1	Detected uncorrectable error (MCR Bit 0)  (Malfunctions Not Necessarily Related to Current Instruction)
2a	Short Warning (MCR Bit 2) C170 IOU Exchange Request (MCR Bit 5) External Interrupt (MCR Bit 8) System Interval Timer (MCR Bit 11) Soft Error Log (MCR Bit 14) Free Flag (UCR Bit 2) Process Interval Timer (UCR Bit 3)  (Tested Between Instructions, Not Generated by Instructions)
2b	Environment Specification Error (MCR Bit 7) † System Call (MCR Bit 10)  Invalid Segment (MCR Bit 12) †† Keypoint (UCR Bit 6) Exponent Overflow (UCR Bit 10) Exponent Underflow (UCR Bit 11) Floating-Point Loss of Significance (UCR Bit 12)  (Tested Between Instructions, Generated by Instructions)
3	Instruction Specification Error (MCR Bit 3) Address Specification Error (MCR Bit 4) Access Violation (MCR Bit 6)  Environment Specification Error (MCR Bit 7) † Page Table Search Without Find (MCR Bit 9)  Invalid Segment (MCR Bit 12) †† Outward Call/Inward Return (MCR Bit 13) Trap Exception (MCR Bit 15) Privileged Instruction Fault (UCR Bit 0) Unimplemented Instruction (UCR Bit 1)
<p>† In group 2b if set by exchange operation. In group 3 if set by call, return, or pop instruction.</p> <p>†† In group 2b if set by load address, return, or pop instructions when RN=0. In group 3 if set by call or trap instructions when RN=0, or by an invalid segment.</p>	

Table 6-3. Interrupt Condition Groups (Sheet 2 of 2)

Group	Interrupt Condition
3	Inter-Ring Pop (UCR Bit 4) Critical Frame Flag (UCR Bit 5) Divide Fault (UCR Bit 7) Debug (UCR Bit 8) Arithmetic Overflow (UCR Bit 9) Floating-Point Indefinite (UCR Bit 13) Arithmetic Loss of Significance (UCR Bit 14) Invalid BDP Data (UCR Bit 15)  (Tested Before Execution, Generated by Instruction)

The PVA from the P register stored in the exchange package during exchange interrupts, or in the Stack Frame Save Area during trap interrupts, points to the instruction shown in table 6-4.

Table 6-4. PVA Instruction Pointer Function

Group	PVA in P Stored During Interrupt
1	Points to the instruction being executed when the malfunction was detected. This instruction does not necessarily initiate the activity that resulted in the malfunction.
2a, 2b	Points to the instruction that would have been executed if the interrupt had not occurred. Therefore, after executing an exchange or return to the interrupted procedure, processing will continue from the PVA stored in exchange package as though the interrupt had not occurred.
3	Points to the instruction that caused the interrupt.

#### ERROR HANDLING

When an error is detected during online CP operations, ICC typically determines at an instruction's point of no return whether the affected instruction should be retried, an exchange or trap interrupt routine initiated, the CP halted, or no immediate remedial action taken (stacking). The decision is based on the nature of the exception or error and the state of associated mask bits. Other factors are whether trap interrupts are enabled and whether a job or monitor process is being executed.

When an interrupt or retry is attempted, all current micrands and instructions are cleared from the pipe. The write operations associated with micrands in ICP Ranks 41 and 50 are inhibited, and a Microtrap microcode routine is addressed in the CST RAM (CST 1.0).

## INSTRUCTION RETRY

The CP flags various parity and SECEDED errors as processor detected malfunctions (PDMs). If sensed before the point of no return, a PDM activates Before PONR MCR Bit 0. Before PONR MCR PDM, in turn, activates Retry in Retry Control, provided three conditions are met - no other exceptions or errors are indicated by the condition registers, Rank 50 PONR is active, and the retry counter is not equal to zero. MAC initially loads the retry counter in Retry Control with a value contained in DEC Bits 52 through 55/56. It decrements each time instruction retry takes place.

Retry enters the Microtrap Address Coder to generate a Microtrap Code. The Microtrap Code addresses a microcode routine in the CST RAM which reloads the affected instruction into the pipeline for reexecuting. If a succession of instruction retries does not eliminate the error and the retry counter decrements to zero, the PDM is then handled by an interrupt routine. If the error is intermittent, the retried instruction may complete execution. In this case the retry is considered successful. PFS Register 84 Bit 41 indicates successful retry.

## INTERRUPTS

The interrupt structure implemented by CP hardware and the operating system is hierarchical. That is, an interrupt may occur in one process which transfers control of the CP to another process. The new process may, in turn, be interrupted to initiate a third process.

There are two types of interrupt mechanisms initiated by error or exception detection circuitry in ICC. They are exchange interrupts and trap interrupts. To determine which is used, the CP must examine the environment in which an arriving error or exception has occurred. In the event neither interrupt is appropriate, the condition may be stacked, that is, stored for future evaluation and processing, or the CP may be halted.

Exchange interrupts store the exchange package currently in the Register File (OPI 1.0) in memory and load a new exchange package into the Register File from memory. An exchange interrupt occurs only in C180 job or C170 job/monitor mode and switches the CP process to C180 monitor mode.

A trap interrupt sends selected exchange package registers from the Register File to a Stack Frame Save Area in memory. Included are minimum of four A Registers and the P Register. The trap interrupt performs an implicit C180 Call Relative (B0) instruction to obtain P of the called trap handler procedure. Trap interrupts may occur in either monitor or job mode. They do not switch the mode.

Mask bits contained in the Monitor Mask Register (MMR) and User Mask Register (UMR) control entry of errors and exceptions from the Before PONR MCR, MCR, Before PONR UCR and UCR into the error handling logic. These mask bits perform logical AND operations with corresponding condition bits in the Condition to Mask Comparators and Interrupt Detection Control. All MCR bits except the state flags are masked. UCR Bits 0 through 6 have their corresponding mask bits permanently set. UCR Bits 7 through 15 may be masked selectively. An example of the basic interrupt mechanism is shown in figure 6-4.

The same procedure applies for Exit Mode Condition Bits 0, 1, which are masked by Exit Mode Mask Register Bits 0, 1 in C170 Exchange Interrupt Control. All mask bits are loaded into the mask registers from the exchange package. Refer to section 1, ICC description, for details of mask bit use in interrupt operations.

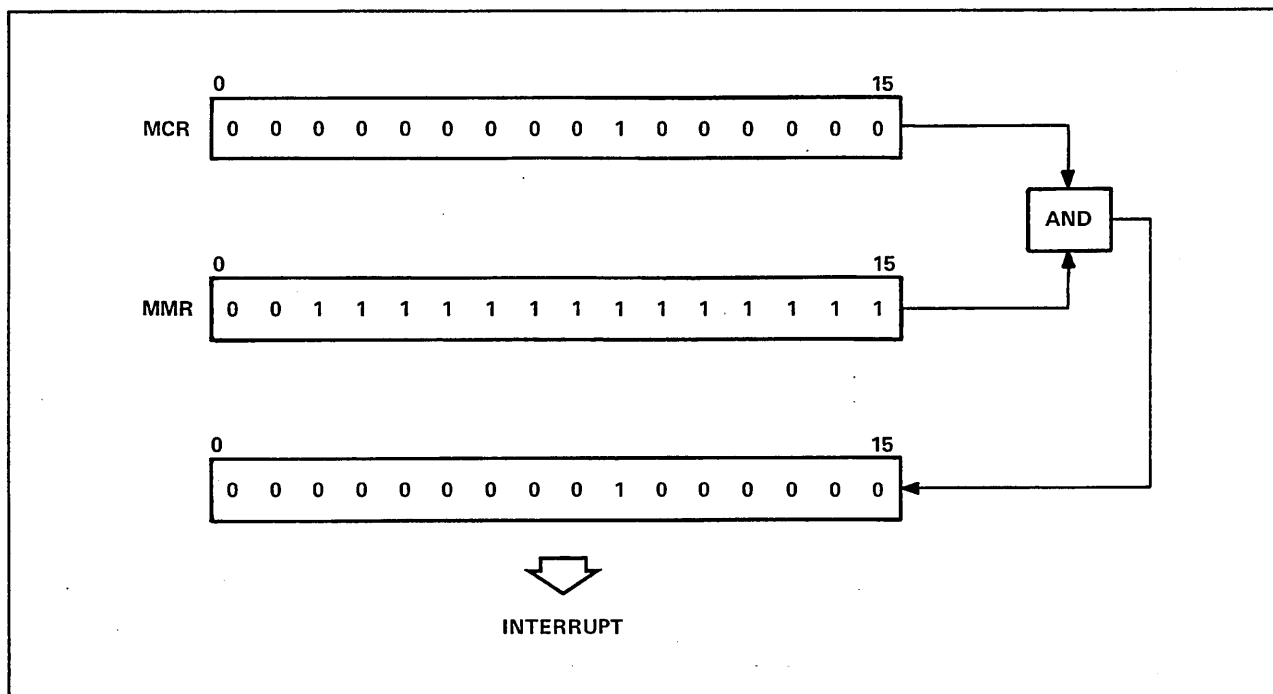


Figure 6-4. Basic Interrupt Mechanism

Tables 6-5 and 6-6 show the actions resulting from detected conditions in various processing environments. Traps Enabled is a condition determined by the exchange package.

The following example is based on information provided by table 6-5. If the CP is in C180 job mode with traps enabled and a Page Table Search Without Find is detected, an exchange interrupt passes CP control to a C180 monitor mode routine. If traps are disabled by the exchange and a Soft Error Log condition activates later, the condition is stacked for future reference. If traps remain enabled after the exchange from job to monitor mode, Soft Error Log triggers a trap interrupt within the monitor mode routine.

Tables 6-5 and 6-6 do not apply when an exchange or trap interrupt is actually in progress. In the exchange interrupt state, a PDM forces the CP to halt. In the trap interrupt state, time-critical MCR conditions in rank 50 interrupt the trap routine by means of an exchange interrupt or halt.

Table 6-5. MCR Interrupts

BIT NUMBER AND DEFINITION		TRAP ENABLED		TRAP DISABLED		WHEN MASK BIT CLEAR
		JOB MODE	MONITOR MODE	JOB MODE	MONITOR MODE	
0	Detected Uncorrectable Error	Mon	EXCH	TRAP	EXCH	HALT
1	Unassigned					
2	Short Warning	Sys	EXCH	TRAP	EXCH	STACK
3	Instruction Specification Error	Mon	EXCH	TRAP	EXCH	HALT
4	Address Specification Error	Mon	EXCH	TRAP	EXCH	HALT
5	Exchange Request	Sys	EXCH	TRAP	EXCH	STACK
6	Access Violation	Mon	EXCH	TRAP	EXCH	HALT
7	Environment Specification Error	Mon	EXCH	TRAP	EXCH	HALT
8	External Interrupt	Sys	EXCH	TRAP	EXCH	STACK
9	Page Table Search Without Find	Mon	EXCH	TRAP	EXCH	HALT
10	System Call	Status - This bit is a flag only and does not cause any hardware action.				
11	System Interval Timer	Sys	EXCH	TRAP	EXCH	STACK
12	Invalid Segment	Mon	EXCH	TRAP	EXCH	HALT
13	Outward Call/Inward Return	Mon	EXCH	TRAP	EXCH	HALT
14	Soft Error Log	Sys	EXCH	TRAP	EXCH	STACK
15	Trap Exception	Status - This bit is a flag only and does not cause any hardware action.				

Table 6-6. UCR Interrupts

BIT NUMBER AND DEFINITION			TRAPS ENABLED		TRAPS DISABLED		WHEN MASK BIT CLEAR
			JOB MODE	MONITOR MODE	JOB MODE	MONITOR MODE	
0 Privileged Instruction Fault	Mon		TRAP	TRAP	EXCH	HALT	These mask bits are permanently set.
1 Unimplemented Instruction	Mon		TRAP	TRAP	EXCH	HALT	
2 Free Flag	User		TRAP	TRAP	STACK	STACK	
3 Process Interval Timer	User		TRAP	TRAP	STACK	STACK	
4 Inter-ring Pop	Mon		TRAP	TRAP	EXCH	HALT	
5 Critical Frame Flag	Mon		TRAP	TRAP	EXCH	HALT	
6 Keypoint	User		TRAP	TRAP	STACK	STACK	
7 Divide Fault	User		TRAP	TRAP	STACK	STACK	STACK
8 Debug	User		TRAP	TRAP	STACK	STACK	STACK
9 Arithmetic Overflow	User		TRAP	TRAP	STACK	STACK	STACK
10 Exponent Overflow	User		TRAP	TRAP	STACK	STACK	STACK
11 Exponent Underflow	User		TRAP	TRAP	STACK	STACK	STACK
12 F. P. Loss of Significance	User		TRAP	TRAP	STACK	STACK	STACK
13 F. P. Indefinite	User		TRAP	TRAP	STACK	STACK	STACK
14 Arithmetic Loss of Significance	User		TRAP	TRAP	STACK	STACK	STACK
15 Invalid BDP Data	User		TRAP	TRAP	STACK	STACK	STACK

Interrupt, halt, or stack determinations occur in the Condition to Mask Comparators, Interrupt Detection Control, Halt Interrupt Control, C170 and C180 Exchange Interrupt Control, and Trap Interrupt Control. Execution of a trap interrupt, exchange interrupt, or a halt requires formation of a CST RAM entry address in the Microtrap Address Coder. Refer to section 3, CST description, for CST RAM addresses associated with interrupts. These conditions also enter Register File Write Enable Control, Clear Pipe Control, and LM Write Abort Control to inhibit Register File writes, clear the instruction pipeline for the interrupt routine, and inhibit memory writes, as required.

Any C180 mode addressing error (MCR Bits 4, 6, 9, and 12) forces capture of the failing address in the Untranslatable Pointer (UTP) Register (ICP).



### Multiple Interrupt Conditions

When more than one bit is set in the MCR and/or UCR, the interrupts are handled in the following order of priority:

1. Halt when any halt condition is present.
2. Exchange when no halt condition is present and any exchange condition is present.
3. Trap when no halt or exchange condition is present and any trap condition is present.
4. Stack when none of the above conditions is present.



SECTION 7

CLOCKS



---

This section and CLK 1.0 describe the system Clock which provides clock signals for IOU, CM, and all of the functional areas in the CP.

The Master Clock Oscillator, the Master Clock Fanout and the 2nd Stage CM Clock Shaper are located on one 210-pak in the Central Memory chassis. (In Models AD112-A/B the Master Clock Oscillator and Master Clock Fanout are located on two 136-paks in the Central Memory chassis; the 2nd Stage CM Clock Shaper is located on a separate Central Memory board.) The Second Stage Clock Fanout is located on five auxiliary boards in the CPU chassis. (MAC shares these boards with CLK.) The remaining circuits are located on a single ZIF board in the CPU chassis.

#### MASTER CLOCK

The Master Clock Oscillator generates a 16-nanosecond (62.5 MHz) Clock signal. It divides the 62.5 MHz to form 32- and 64- nanosecond clock signals.

#### CLOCK DISTRIBUTION

The Master Clock Fanout sends:

- 32- and 64-nanosecond Clock to IOU.
- 16-nanosecond Clock to CP through First and Second Clock Fanouts.
- 16-nanosecond Clock to CP through CM 32-nanosecond Clock Generator as 32-nanosecond Clock.
- 16- and 64-nanosecond Clocks to CP which forms Phase Maker.

#### PHASE MAKER

The Phase Maker generates a 16-nanosecond pulse every major cycle (64 nanoseconds) to synchronize the Phase Generators in the CP functional areas. The combination of 16- and 64-nanosecond Clock produces the Phase Maker. The Phase Maker and the next 16-nanosecond Clock produces Phase 0 in each Phase Generator. Therefore, Phase 0 and subsequent Phases 1, 2, and 3 are synchronized throughout the CP.

#### SECOND STAGE CLOCK FANOUT

This circuit generates four 8-nanosecond pulses every major cycle (64 nanoseconds) for use by the CP functional areas. The combination of Phases 0 through 3 from SM and 16-nanosecond Clock produces Time 0 (T0) through Time 3 (T3) clocks. This circuit also distributes Clock every minor cycle (16 nanoseconds) to the CP functional areas.

#### MAINTENANCE FEATURES

The Clock includes the following maintenance features.

- Switch on Master Clock Oscillator for either normal or clock tuning mode.
- Feedback paths for clock tuning.
- Switch on Master Clock Oscillator to vary clock frequency by  $\pm 2$  percent.
- Wide, Narrow Margins from the Dependent Environmental Control (DEC) Register in MAC to vary 16-nanosecond Clock pulse width by  $\pm 10$  percent.

Comments (continued from other side)

Please fold on dotted line;  
seal edges with tape only.

FOLD

OLD

FOLD



NO POSTAGE  
NECESSARY  
IF MAILED  
IN THE  
UNITED STATES

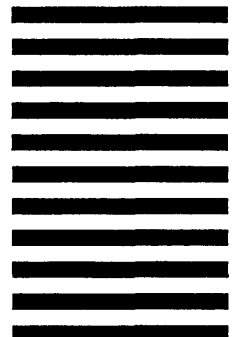
**BUSINESS REPLY MAIL**

First-Class Mail Permit No. 4760 St. Paul, MN

POSTAGE WILL BE PAID BY ADDRESSEE

**CONTROL DATA**

Technical Publications ARH219  
4201 Lexington Avenue N.  
St. Paul, MN 55126-9983



## COMMENT SHEET

CDC CYBER 170 Models 845 and 855, CYBER 180 Models 840,  
845, 850, 855, and 860, CYBER 840A, 850A, 860A, and 870A  
**MANUAL TITLE:** Central Processor and Central Memory  
Hardware Maintenance Manual

**PUBLICATION NO.:** 60458170

**REVISION:** F

**NAME:** \_\_\_\_\_

**COMPANY:** \_\_\_\_\_

**STREET ADDRESS:** \_\_\_\_\_

**CITY:** \_\_\_\_\_ **STATE:** \_\_\_\_\_ **ZIP CODE:** \_\_\_\_\_

This form is not intended to be used as an order blank. Control Data Corporation welcomes your evaluation of this manual. Please indicate any errors, suggested additions or deletions, or general comments below (please include page number references).

☐ Please Reply

☐ No Reply Necessary

NO POSTAGE STAMP NECESSARY IF MAILED IN U.S.A.

FOLD ON DOTTED LINES AND TAPE





